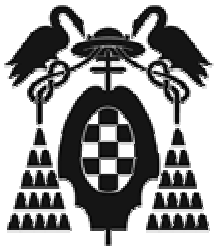

UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

Departamento de Automática

OPTIMIZACIÓN EVOLUTIVA DE LOS PARÁMETROS DE CONTROL DE UN ALGORITMO GENÉTICO

TESIS DOCTORAL

José Ángel Fernández Prieto

Alcalá de Henares, 2009

Escuela Politécnica Superior

Departamento de Automática

TESIS DOCTORAL

OPTIMIZACIÓN EVOLUTIVA DE LOS PARÁMETROS DE CONTROL DE UN ALGORITMO GENÉTICO

Autor:

José Ángel Fernández Prieto

Ingeniero de Telecomunicación

Directores de tesis:

Dr. D. Juan Ramón Velasco Pérez

Doctor Ingeniero de Telecomunicación

Dr. D. Luis Magdalena Layos

Doctor Ingeniero de Telecomunicación

Alcalá de Henares, 2009

A mi familia. En especial a mis padres, Alfonso y Ana;
a mi esposa, Celia; y a mis dos hijos, Javier y Paula,
con los que tengo pendiente recuperar el tiempo que
últimamente no les he podido dedicar.

II

Agradecimientos

Tras la finalización de este trabajo de investigación, llega el momento de mirar hacia atrás y agradecer, aunque sólo sea con unas sinceras palabras, a todas aquellas personas que, de una manera u otra, lo han hecho posible.

En primer lugar, quiero agradecer a Juan Ramón Velasco y a Luis Magdalena el haberme dado la posibilidad de realizar esta tesis cuando en los comienzos de la Universidad de Jaén no existía infraestructura para ello. Además, también quiero reconocerles la confianza depositada en mi, la ayuda que me han prestado así como su disponibilidad siempre que los he requerido.

También desearía dar las gracias a Francisco Herrera, por las valiosas recomendaciones que me ha dado en los distintos congresos en los que hemos coincidido.

No me olvido de mis compañeros del Departamento de Ingeniería de Telecomunicación de la Universidad de Jaén, a los que también quiero dar mi agradecimiento, en especial a: Manuel Angel, Joaquín, Seba y Enrique, con los que he compartido numerosos viajes buscando el mismo objetivo; a Curro, compañero de despacho, siempre dispuesto a echar una mano; y a Antonio, por su ayuda en la generación del tráfico de poisson.

Y, finalmente, a mi familia; en especial, a mis padres, por el empuje y consejos que siempre me han dado; a Celia, por su constante apoyo; a Javier y Paula, a los que ya puedo contestarles que se ha acabado ese trabajo tan largo.

Sé que dejo de nombrar a muchos compañeros y amigos. También a vosotros os doy las gracias.

Gracias a todos.

Resumen

El comportamiento de un Algoritmo Genético viene determinado, en gran medida, por los parámetros que utiliza, como son: el tamaño de la población y las probabilidades de selección, cruce y mutación. Sin embargo, no existe una regla general mediante la cual se puedan seleccionar los parámetros apropiados para cada tipo de problema. En unos casos, se utilizan los valores recomendados en la literatura, mientras que en otros, su elección representa un problema de prueba y error. Además, distintos autores argumentan que estos valores no deben ser fijos durante la ejecución del algoritmo ya que es un proceso intrinsecamente dinámico y adaptativo.

En esta tesis doctoral se propone un sistema de optimización de parámetros que combina dos de las técnicas recogidas en la literatura para mejorar el comportamiento de un Algoritmo Genético: la meta-evolución y la adaptación de parámetros. Con el objeto de validar el sistema propuesto, este ha sido aplicado sobre los siguientes Algoritmos Genéticos, los cuales utilizan distintos tipos de codificación:

1. Algoritmo Genético con codificación binaria, con el objetivo de minimizar un conjunto de seis funciones representativas.

2. Algoritmo Genético con una codificación híbrida, binaria y real, de un sistema borroso-genético basado en el enfoque de Pittsburgh.

3. Algoritmo Genético con codificación real, el cual se encuentra integrado con un simulador de redes de comunicaciones, con el objeto de comprobar el funcionamiento en un sistema real: un protocolo de comunicaciones en una red.

Por último, se ha comprobado el comportamiento de los distintos algoritmos al utilizar los parámetros hallados por el sistema propuesto. Los resultados se han comparado con los obtenidos por los principales métodos de adaptación de parámetros. Además, se han llevado a cabo diversos tests estadísticos para averiguar si existen diferencias significativas entre los resultados obtenidos.

Abstract

The behavior of a Genetic Algorithm is determined by the control parameter settings, such as selection probability, crossover probability, mutation probability and population size. Nevertheless, there are no standard rules for choosing appropriate values for these parameters. This decision is usually taken in terms of the most common values or experimental formulas given in literature, or by means of trial an error methods. Furthermore, any authors give arguments that any static set of parameters, having the values fixed during an Genetic Algorithm, seems to be inappropriate; a Genetic Algorithm is an intrinsically dynamic, adaptive process.

This Ph. D. Thesis propose an approach based on a meta-level GA combined with an adaptation strategy of the GA control parameters to find and adjust the best control parameters to improve the GA performance. In order to validate the approach, it have been applied to three Genetics Algorithms which use different coding schemes to solve different types of optimization problems, such as:

1. a binary-coded Genetic Algorithm. The objective of this algorithm is to minimize six frequently used test functions.

2. a hybrid-coded (binary and real) Genetic Algorithm, which belong to a Genetic Fuzzy Rule-Based System based on the Pittsburgh approach.

3. a real-coded Genetic Algorithm, which is integrated with a network simulator to verify the network protocol performance. In this case, the algorithm aims at generating the worst case traffic for the protocol under analysis.

Finally, different comparisons are performed, aiming to assess the acceptable optimization power of the proposed system. The results have been compare with the ones obtained for other methods for changing the values of parameters. Moreover, a statiscal analysis have been done to ascertain if differences are significant between the proposed system and the other algorithms.

Índice general

Índice de figuras

Índice de tablas

Parte I

Planteamiento de la Investigación y Revisión de Conocimientos

Capítulo 1

Introducción

1.1. Contexto y localización de la investigación

Los Algoritmos Genéticos simulan el proceso evolutivo darwiniano en sistemas artificiales. J. Holland [Holland, 1975] fue uno de los pioneros en mostrar como aplicar, para la resolución de problemas, los procesos evolutivos a los sistemas artificiales, diseñados a partir de los sistemas naturales. Así, se definen los Algoritmos Genéticos como algoritmos de búsqueda, basados en la evolución natural y la genética, que proporcionan un mecanismo de búsqueda y optimización robusto en sistemas complejos, ofreciendo una vía útil para problemas de optimización que requieran eficiencia y eficacia [Goldberg, 1989, Michalewicz, 1992].

Uno de los aspectos más importantes dentro de la estructura de un Algoritmo Genético son los valores de los parámetros que utiliza, como son: el tamaño de la población, la probabilidad de selección, la probabilidad de cruce y la probabilidad de mutación, puesto que su elección determinará, en gran medida, que el algoritmo encuentre una solución óptima, y de forma eficiente, al problema dado.

Para que un Algoritmo Genético obtenga un rendimiento satisfactorio es necesario configurarlo, de forma específica, para cada tipo de problema [Bäck, 1992], dado que no existe una regla general mediante la cual se puedan hallar unos parámetros apropiados aplicables a todo tipo de problemas. Sin embargo, en la mayoría de los casos se utilizan los valores recomendados en la literatura, y en otros, su elección representa un problema de prueba y error. De este modo, la búsqueda de los valores de los parámetros que optimizan el comportamiento de un AG ha sido objeto de

diversos estudios desde los primeros trabajos, y sigue teniendo completa actualidad: [DeJong, 1975, Grefenstette, 1986, Schaffer et al., 1989, Bramlette, 1991, Srinivas y Patnaik, 1994, Wu y Chow, 1995, Eiben et al., 1999, Cao y Wu, 1999, Cicirello y Smith, 2000, Herrera y Lozano, 2003, Subbu y Bonissone, 2003, Sahin y Abbate, 2004, Jansen et al., 2005, Eiben et al., 2007, Meyer-Nieberg y Beyer, 2007, Cantú-Paz, 2007, DeJong, 2007, Sun, 2007, Tsai y Chao, 2008, Vajda et al., 2008].

Por otro lado, diferentes valores de los parámetros pueden ser los óptimos en distintos estados del proceso evolutivo, razón por la cual algunos autores han sugerido la utilización de técnicas adaptativas, donde se modifica la configuración del algoritmo en función de la información que se tenga sobre el espacio de búsqueda. En [Hesser y Männer, 1991, Bäck, 1991, 1992, Eiben et al., 1999] se argumenta que la utilización de valores fijos durante la ejecución del algoritmo es inapropiada, ya que se trata de un proceso intrínsecamente dinámico, adaptativo y, por tanto, el uso de parámetros fijos que no cambian su valor en la ejecución va en contra de ese espíritu y puede conducir a un menor rendimiento.

Así surgieron los Algoritmos Genéticos Adaptativos, que ajustan dinamicamente los componentes de un Algoritmo Genético durante su ejecución, es decir, configuran el algoritmo mientras se resuelve el problema. Su principal objetivo es ofrecer un comportamiento más apropiado, tanto en la exploración como en la explotación de la información, para evitar una convergencia prematura en el problema y mejorar el resultado final. Uno de los enfoques consiste en adaptar los parámetros del Algoritmo Genético. Los principales métodos de adaptación de los parámetros se pueden clasificar en tres categorias:

- Control determinístico, donde los valores de los parámetros son modificados atendiendo a alguna regla determinística sin utilizar ningún tipo de realimentación del estado de la búsqueda.
- Control adaptativo. Con este método los valores de los parámetros son modificados atendiendo al estado de la búsqueda de la solución al problema, existiendo, por tanto, una realimentación que es utilizada para determinar la magnitud del cambio.
- Control auto-adaptativo, donde los parámetros que se desean adaptar son codificados dentro de los individuos de la población y evolucionan junto a ellos, aplicándoles también los operadores de cruce y mutación.

Además, estas adaptaciones pueden realizarse en distintos niveles dependiendo del ámbito donde se produzcan, por lo que es posible distinguir:

- Adaptación a nivel de la población, cuando se ajustan parámetros globales cuyos valores afectan a todos los individuos de la población.
- Adaptación a nivel del individuo, cuando se ajustan parámetros en un individuo y cuyos valores afectan únicamente a ese individuo.
- Adaptación a nivel del componente, cuando se ajustan parámetros locales que afectan a algún gen o genes de un individuo en la población.

Por tanto, como puede apreciarse, son múltiples las combinaciones que se pueden dar a la hora de configurar un algoritmo en cuanto a valores de los parámetros a utilizar, si fijos o adaptativos y, en este último caso, nivel al cual realizar la adaptación. De esta forma, se hacen necesarios métodos ó técnicas eficientes que ayuden a encontrar, para cada problema, los valores apropiados de los parámetros y determinar, en cada caso, si realizar una adaptación o no de los mismos.

1.2. Objetivos

El propósito principal de la presente tesis doctoral es *diseñar, desarrollar y validar un sistema de optimización de parámetros de un Algoritmo Genético*. El objetivo tiene como finalidad realizar una propuesta de una técnica de optimización de parámetros que permita, para cada tipo de problema, hallar un conjunto de parámetros que optimice el comportamiento de un Algoritmo Genético.

Este objetivo principal nos lleva a la propuesta de 3 objetivos adicionales subordinados:

- Portabilidad. El sistema podrá ser utilizado en distintos escenarios, con independencia del tipo de codificación utilizado en el Algoritmo Genético.
- Independencia del problema. El sistema debe ser totalmente independiente del problema a resolver a por el Algoritmo Genético.
- Flexibilidad. El sistema deberá ser flexible en cuanto a los parámetros que se deseen optimizar. A su vez, ofrecerá la posiblidad de que dichos parámetros

sean fijos o adaptativos durante la ejecución del Algoritmo Genético. En este sentido, se diseñará un nuevo método de adaptación de parámetros, donde el sistema, además de hallar los parámetros que optimizan el comportamiento del Algoritmo Genético, deberá encontrar la forma en la que estos deben variar a lo largo de la ejecución del mismo.

1.3. Metodología

Una vez especificado el objetivo principal de la tesis doctoral, la metodología a seguir será la siguiente:

1. Estudios iniciales.

   Se realizará un estudio del estado del arte tanto de los Algoritmos Genéticos como de los Algoritmos Genéticos Adaptativos.

2. Diseño y desarrollo del sistema de optimización.

   Se diseñará un sistema para encontrar los valores de los parámetros que optimizan el comportamiento de un AG, con independencia del tipo de codificación utilizado en el AG y del problema a resolver. También se diseñará un método de adaptación de los valores de los parámetros donde el sistema, además de encontrar los valores "óptimos", encontrará la forma en la que estos deben variar a lo largo del tiempo.

3. Desarrollo e implementación de distintos escenarios de aplicación.

   Este punto se centra en el desarrollo e implementación de los escenarios donde posteriormente será aplicado el sistema de optimización. Se implementarán tres escenarios distintos, en cada uno de los cuales existirá un Algoritmo Genético con una codificación diferente. En concreto, serán:

   - un Algoritmo Genético con codificación binaria, para la resolución de un problema de minimización sobre un conjunto de seis funciones representativas, utilizadas frecuentemente en la literatura como funciones de evaluación del comportamiento de los Algoritmos Genéticos.
   - un Algoritmo Genético con una codificación híbrida, binaria y real. En este caso, se trata de un sistema borroso-genético, basado en el enfoque de Pittsburgh, donde se realiza un aprendizaje evolutivo de una base de conocimiento sobre un controlador borroso.

- un Algoritmo Genético con codificación real. En este caso, se deberá implementar un sistema compuesto por un simulador de redes de comunicaciones y un algoritmo con codificación real, con el objeto de comprobar el funcionamiento de protocolos de comunicaciones en una red.

En cada escenario, además de existir distintas codificaciones en el Algoritmo Genético, se implementarán distintos métodos de selección y de sustitución, asi como distintos operadores de cruce y mutación.

4. Integración del sistema de optimización con los escenarios implementados.

   Una vez diseñado el sistema de optimización e implementado los distintos escenarios, se procederá a la integración de ambas partes para, posteriormente, realizar la validación. Tras realizar la integración, se obtendrán los parámetros que optimizan el comportamiento del algoritmo en cada caso.

5. Validación del sistema de optimización.

   Por último, con el objeto de validar el sistema propuesto, se realizarán diversos experimentos en cada uno de los escenarios. En primer lugar, se comprobará el comportamiento del Algoritmo Genético con los parámetros hallados por el sistema propuesto. Posteriormente, se compararán los resultados con los obtenidos al utilizar los principales métodos de adaptación de los parámetros, los cuales, a su vez, deberán ser implementados previamente. Para llevar a cabo esta comparación, se ejecutarán todos los algoritmos en 30 ocasiones y, seguidamente, se realizarán distintos tests estadísticos.

1.4. Estructura de la tesis

El presente documento se estructura en tres bloques temáticos, compuestos a su vez por una serie de capítulos.

- **Planteamiento de la Investigación y Revisión de Conocimientos.**

  Este bloque temático está compuesto de los siguientes capítulos:

  Capítulo primero, que se corresponde con esta introducción, en el que se presentan los objetivos de la investigación y la estructura de la tesis doctoral.

Capítulo segundo, donde se realiza una revisión del estado del arte en relación a los Algoritmos Genéticos. Se presentan los conceptos básicos de funcionamiento, los modelos básicos de implementación, sus estructuras, así como los componentes de los que constan. A su vez, se muestran distintas soluciones propuestas en la literatura para mejorar el comportamiento de este tipo de algoritmos.

Capítulo tercero, el cual está dedicado a la revisión de los conceptos más importantes sobre un caso particular de Algoritmos Genéticos, los Algoritmos Genéticos Adaptativos. En primer lugar, se muestra una clasificación de los mismos, basada en: a) los componentes que pueden ser adaptados; b) los niveles a los que se puede producir la adaptación y, c) los tipos en los que se realiza dicha adaptación. A continuación, se describen las contribuciones más importantes recogidas en la literatura sobre la adaptación de uno de los componentes de un Algoritmo Genético: los parámetros de control.

Capítulo cuarto, donde se presentan los fundamentos de los sistemas borrosos y borroso-genéticos, destacando la estructura de un controlador borroso así como los principales enfoques existentes en el aprendizaje de los sistemas borroso-genéticos.

- **Desarrollo de la Investigación.**

  Este bloque se compone de los siguientes capítulos:

  Capítulo quinto, donde se describe el sistema de optimización de parámetros propuesto cuyo objetivo es hallar los parámetros que mejoran el comportamiento de un Algoritmo Genético. Se presenta la estructura del sistema y se detalla cada uno de sus componentes.

  Capítulos sexto, séptimo y octavo, en los que se describen cada uno de los escenarios donde ha sido aplicado el sistema de optimización propuesto. A su vez, en cada capítulo, se muestran los experimentos realizados así como los resultados obtenidos en cada uno de ellos.

- **Conclusiones y Líneas Futuras.**

  Por último, este bloque se compone de un único capítulo, el *capítulo noveno*, donde se presentan las conclusiones extraídas durante el desarrollo de la tesis así como las posibles nuevas líneas de investigación que pueden surgir de la misma.

Capítulo 2

Algoritmos Genéticos

2.1. Introducción

En este capítulo se proporciona, en primer lugar, una visión general de la computación evolutiva, describiendo brevemente las diferentes variantes de los algoritmos evolutivos. Posteriormente, se presentan los conceptos básicos de funcionamiento de un Algoritmo Genético, su estructura, así como los componentes de los que consta. Por último, se aborda el problema de la convergencia vs. a diversidad y se muestran algunas soluciones, propuestas en la literatura, para mejorar su comportamiento.

La computación evolutiva recoge las distintas vertientes en el diseño de los algoritmos evolutivos como herramientas para el análisis de sistemas no lineales complejos por medio de simulación computacional y basados en los principios de los procesos de evolución natural [Herrera et al., 1995a]. Por tanto, se puede definir la computación evolutiva como una familia de modelos computacionales ó técnicas heurísticas basadas en los principios de la evolución natural. Estas técnicas heurísticas pueden clasificarse en cuatro principales categorías, que fueron desarrolladas en distintos lugares y con distintas motivaciones:

- Algoritmos Genéticos.
- Estrategias de Evolución.
- Programación Evolutiva.
- Programación Genética.

2.2. Evolución natural. Computación Evolutiva

La selección natural, concepto introducido por Charles Darwin en su libro *On the Origin of Species By Means of Natural Selection* [Darwin, 1859], es el proceso a través del cual los organismos que mejor se adaptan a un entorno desplazan a los menos adaptados mediante la acumulación de cambios genéticos favorables en la población a lo largo de las distintas generaciones. Darwin expuso que una especie que no sufriera cambios se volvería incompatible con su entorno ya que éste tiende a cambiar con el transcurrir del tiempo. También destacó las similitudes entre padres e hijos observadas en la naturaleza y que ciertas caracteristicas de las especies eran hereditarias. Además, indicó que de generación a generación ocurren cambios para hacer al nuevo individuo más apto para sobrevivir.

Durante millones de años las diferentes especies se han tenido que adaptar para poder sobrevivir a un entorno cambiante, por lo que la adaptación es esencial para la supervivencia de los individuos. En la naturaleza, los procesos evolutivos se producen cuando se dan los siguientes condicionantes [Herrera et al., 1995a]:

- un individuo tiene la habilidad de reproducirse.
- existe una población de individuos que son capaces de reproducirse.
- existe alguna diferencia entre los individuos que se reproducen.
- algunas diferencias en la habilidad para sobrevivir en el entorno están asociadas con esa variedad.

La diversidad de individuos en la población se debe a que cada individuo posee unos cromosomas diferentes a los demás y, por tanto, cada individuo se comporta de forma diferente en el entorno. Los cambios en la estructura y comportamiento de cada individuo se van a reflejar en su grado de supervivencia, adaptación y en el nivel de reproducción.

Los individuos mejor adaptados a su entorno tendrán mayores probabilidades de sobrevivir y de generar una mayor descendencia. Por el contrario, los individuos peor adaptados tendrán una menor descendencia, o incluso nula, lo cual implica que los genes de los individuos mejor adaptados se trasladarán a un número cada vez mayor de individuos de las sucesivas generaciones. En ocasiones, la combinación de las buenas caracteristicas de los padres puede originar que la descendencia esté mejor

adaptada al medio que los mismos padres. Además, mediante las mutaciones se producen alteraciones en las informaciones genéticas de los individuos, introduciendo nuevas variaciones genéticas y causando que la descendencia posea material genético diferente a los padres. De esta manera, las especies evolucionan adaptándose mejor al medio mientras transcurren las generaciones. Pero la adaptación de un individuo al medio no sólo está determinada por su composición genética, sino que también influyen otros factores como el aprendizaje, en unas ocasiones adquirido por el método de prueba y error, y en otras, adquirido por imitación del comportamiento de los padres.

Así, imitando la mecánica de la evolución biológica en la naturaleza, los Algoritmos Evolutivos (AE) copian los procesos que tienen lugar en la selección natural para la resolución de problemas computacionales, adoptando una terminología similar a la biológica para describir sus elementos estructurales y operaciones algorítmicas.

En términos biológicos, todas las células de un organismo vivo contienen información hereditaria codificada en moléculas de ácido desoxirribonucleico (ADN); esta información dirige la actividad de la célula y asegura la reproducción y el paso de las caracteristicas a la descendencia. Dentro del núcleo de cada célula, las moléculas de ADN están organizadas en cromosomas que suelen aparecer dispuestos en pares idénticos. El ADN del interior de cada cromosoma es una molécula única muy larga y arrollada que contiene secuencias lineales de genes[1]. A la combinación especifica de genes en un individuo se le denomina *genotipo*, mientras que el *fenotipo* es la manifestación visible del genotipo (rasgos físicos del individuo) y el término *alelo* describe el valor de un gen.

En la terminología de los AE se utilizan alternativamente los términos cromosoma y genotipo para describir a un conjunto de parámetros que codifican una posible solución al problema. El término gen hace referencia a una particular entidad funcional de la solución, por ejemplo, a un parámetro especifico en un problema de optimización. La *adaptación* del individuo se corresponde con la *calidad* de la posible solución mientras que el *entorno*, se corresponde con el *problema a resolver*. En la figura 2.1 se muestra la similitud entre las dos terminologías.

[1]Un gen es una entidad funcional que codifica una caracteristica especifica del individuo, p.e., el color del pelo.

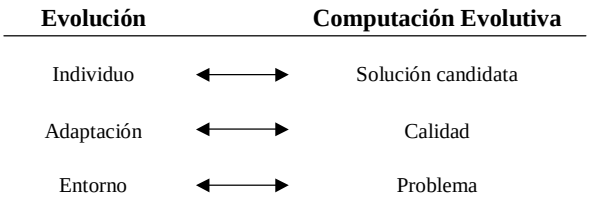

Figura 2.1: *La metáfora. Evolución natural vs. Algoritmos Evolutivos.*

Los AE trabajan con una población de cromosomas, cada uno representando una posible solución a un problema dado. A cada cromosoma se le asigna una puntuación de adaptación, dependiendo de cómo de buena ha sido su respuesta al problema, a través de una función de evaluación (que suele ser la función objetivo a optimizar). Teniendo en cuenta el principio de Darwin, los cromosomas más adaptados (aquellos que han obtenido una mayor puntuación) son los que tienen mayor probabilidad de ser seleccionados como padres, produciendo descendientes en la próxima generación. Los cromosomas menos adaptados poseen pocas probabilidades de que sean seleccionados para la reproducción y desaparecen. Los operadores genéticos de cruce y mutación son aplicados a los padres para generar nuevas posibles soluciones. De esta forma, a lo largo de varias generaciones, como resultado de este ciclo evolutivo de selección, cruce y mutación, se encontrarán mejores soluciones dentro de la población.

La figura 2.2 muestra la estructura de un AE genérico. Un AE es un algoritmo estocástico que mantiene una población de cromosomas, $P(t) = \{x_1^t, ..., x_N^t\}$, durante la generación t. La población se inicializa en el instante t=0, por ejemplo, de forma aleatoria. Cada posible solución, x_i^t, es *evaluada* a través de una función de evaluación f, dando $f(x_i^t)$ una medida de su nivel de adaptación. Seguidamente, un conjunto de padres es *seleccionado* de la población $P(t)$, de forma que los cromosomas mejor adaptados tengan una probabilidad mayor de poder reproducirse. El *cruce* itercambia la información genética entre dos cromosomas para formar un descendiente, que incorpora caracteristicas de sus padres. El descendiente, posteriormente, es sometido a una *mutación* que alterará de forma aleatoria sus genes. Por último, la población actual es *reemplazada* con la nueva descendencia creada, formando la siguiente generación. El ciclo evolutivo de evaluación, selección, cruce, mutación y sustitución continúa hasta alcanzar los criterios de finalización, los cuales pueden ser definidos, por ejemplo, como un número máximo de generaciones o cuando se ha alcanzado un cierto nivel de adaptación por parte del mejor cromosoma.

Inicio

- t=0;
- Inicialización de la población P(t).
- *Evaluación* de la población P(t).
- **Mientras** (no se cumplan los criterios de finalización)

  Selección de los padres en la población P(t).
  Cruce de los padres para obtener la descendencia.
  Mutación de la descendencia.
  Sustitución de la población con la descendencia.
  t=t+1;
  Evaluación de la población P(t).

Final

Figura 2.2: *Estructura de un Algoritmo Evolutivo.*

Todas las técnicas heurísticas que recoge la computación evolutiva (Algoritmos Genéticos, Estrategias de Evolución, Programación Evolutiva y Programación Genética) están basadas en los mismos principios indicados anteriormente; sin embargo, utilizan diferentes formas de representación de los cromosomas, distintas técnicas de selección y sustitución, así como distintos operadores genéticos de cruce y mutación.

2.2.1. Algoritmos Genéticos

Los Algoritmos Genéticos (AGs) simulan el proceso evolutivo darwiniano en sistemas artificiales y su desarrollo se debe en gran parte a John Holland, investigador de la Universidad de Michigan, que a principios de la década de los 70 desarrolló una técnica que imitaba en su funcionamiento a la selección natural. En un principio, esta técnica recibió el nombre de *planes reproductivos*. En su libro *Adaptation in Natural and Artificial Systems* [Holland, 1975] mostró como aplicar los procesos evolutivos a los sistemas artificiales, diseñados a partir de los sistemas naturales, para la resolución de problemas. A partir de su publicación, los *planes reproductivos*

pasaron a denominarse lo que hoy se conoce como AGs. Así, se definen los AGs como algoritmos de búsqueda, basados en la evolución natural y la genética, que proporcionan un mecanismo de búsqueda y optimización robusto en sistemas complejos y ofrecen una vía útil para problemas de optimización que requieran eficiencia y eficacia [Goldberg, 1989, Michalewicz, 1992]. Los AGs operan sobre una población de individuos que, generalmente, son representados como cadenas binarias de longitud fija[2]. Como resultado de la aplicación de los operadores genéticos sobre la población, ésta evoluciona hacia una nueva generación donde los individuos mejor adaptados al problema (los más cercanos al valor óptimo buscado) sustituyen a aquellos peor adaptados.

2.2.2. Estrategias Evolutivas

Las Estrategias Evolutivas (EE), basadas también en los principios de los procesos evolutivos naturales, fueron desarrolladas a mediados de los años 60 en Alemania para resolver problemas hidrodinámicos de alta complejidad. Se aplican fundamentalmente a problemas de optimización numérica y, en este caso, los cromosomas son vectores de números reales. La primera versión del algoritmo, denominado (1+1)-EE, utilizaba un único padre, a partir del cual se generaba un único hijo (aplicando una perturbación sobre el padre se generaba el hijo). Posteriormente, ambos competían entre sí, sobreviviendo el más adaptado, por tanto, el peor individuo tenía una probabilidad cero de sobrevivir. En la figura 2.3 se muestra la estructura del algoritmo (1+1)-EE, donde N(0,σ) es una distribución normal con media cero y desviación estándar σ.

En 1973, Rechemberg introdujo las bases de las EE en su obra *Evolutionsstrategie: Optimierung Technisher Systeme nach Prinzipien der Biologischen Evolution* [Rechenberg, 1973]. Propuso una variante al algoritmo original denominada estrategia evolutiva $(\mu+1)$-EE, donde μ es el número de padres utilizados para generar un único hijo, el cual reemplazaba al peor padre de la población; solamente existía selección y mutación. A su vez, estableció una regla para ajustar la desviación estándar durante el proceso evolutivo.

[2]Aunque posteriormente se han diseñado AGs con codificación real.

Inicio

- i=0;
- Creación de un individuo de forma aleatoria $\vec{x}_i$
- **Mientras** (no se cumplan los criterios de finalización)

  Obtención de un valor Z=N(0,σ)

  Generación del nuevo individuo $\vec{y}_i = \vec{x}_i + Z$

  Evaluación del nuevo individuo $\vec{y}_i$

  Si la adaptación de $\vec{y}_i$ es mayor a la de $\vec{x}_i$ entonces $\vec{x}_{i+1} = \vec{y}_i$

  En caso contrario $\vec{x}_{i+1} = \vec{x}_i$

  i=i+1;

Final

Figura 2.3: *Estructura de un algoritmo (1+1)-EE.*

Dicha regla, conocida como 'regla del éxito 1/5', indica que:

> *La relación entre mutaciones exitosas* [3] *y el total de mutaciones debería ser exactamente 1/5. Si es mayor, entonces debería incrementarse la desviación estándar. Si es menor, entonces debería decrementarse.*

Posteriormente, en 1981 Schwefel [Schwefel, 1981] introdujo el concepto de recombinación y modificó la estrategia $(\mu + 1)$-EE, dando lugar a las estrategias denominadas:$(\mu + \lambda)$-EE y (μ, λ)-EE, donde λ es el número de hijos producidos por los padres. En ambas, un par de padres generan a un hijo mediante el cruce, el cual, posteriormente, es perturbado mediante la mutación. La diferencia radica en que en la primera, un total de μ padres producen λ hijos (manteniéndose la población en μ individuos). Los μ mejores individuos obtenidos de la unión de padres e hijos sobreviven. Por tanto, los padres sobreviven hasta que son reemplazados por hijos mejores que ellos. En la segunda, la descendencia reemplaza directamente a los padres, sin hacer ningún tipo de comprobación, sobreviviendo unicamente los μ mejores hijos. En 1996, Thomas Bäck derivó una regla de éxito 1/7 para (μ, λ)- EE.

[3] Una mutación es considerada como exitosa si se produce una descendencia que es mejor que los padres

En las EE, además de evolucionar las variables del problema, se pueden hacer evolucionar los parámetros que utiliza la estrategia: la desviación estándar. A esto se le conoce como auto-adaptación y permite al algoritmo ajustar por sí mismo sus parámetros.

2.2.3. Programación Evolutiva

La Programación Evolutiva (PE) fue propuesta por L.J. Fogel en sus trabajos *Autonomous automata* [Fogel, 1962] y *Artificial Intelligence Through Simulated Evolution* [Fogel et al., 1966] con el objetivo inicial del diseño automático de sistemas inteligentes, siendo su trabajo continuado por su hijo D.B. Fogel en los años 80 [Fogel, 1991]. Tradicionalmente no ha existido ninguna idea preconcebida de cómo representar a los individuos, aunque inicialmente fueron representados mediante máquinas de estado finito. En cada caso concreto, se deberá elegir una representación que se adapte convenientemente al problema, pudiéndose utilizar estructuras como: vectores de números reales, listas ordenadas de nodos, etc.

En la actualidad, la PE comparte muchas caracteristicas con las EE, como por ejemplo, la representación de los individuos como números reales. Así, los algoritmos de la PE son aplicados a la resolución de problemas de optimización numérica mediante la aplicación de múltiples operadores de mutación y con todo tipo de representaciones. Generalmente, se aplica la autoadaptación de los parámetros de forma similar a lo realizado en las EE. Sin embargo, en la PE no se emplea el cruce, debido a que las formas de mutación utilizadas son muy flexibles y pueden producir perturbaciones similares. Una vez que todos los individuos, N, se han inicializado, todos se seleccionan para ser padres, que siendo mutados, producen N hijos. Los hijos son evaluados junto a los padres y N (de los 2N) individuos sobreviven, utilizando una función probabilistica basada en la función de adaptación[Herrera et al., 1995a].

2.2.4. Programación Genética

La Programación Genética (PG) es utilizada para la generación automática de programas de ordenador y puede considerarse como un caso particular de AGs. Fue propuesta por John Koza [Koza, 1992] a principios de la decada de los 90 y trata de encontrar el programa que resuelva un determinado problema, efectuándose un

proceso de búsqueda sobre el espacio de posibles programas. Los individuos que componen la población son programas estructurados en forma jerárquica, con formas y tamaños diferentes que cambian dinámicamente a lo largo del proceso. Por tanto, los cromosomas codifican, de algún modo, programas de ordenador.

Existen distintas formas de representación de los programas pero la más utilizada es la de árbol de expresiones. Esta representación se basa en la existencia de una gramática libre de contexto que define las sentencias válidas del lenguaje. El cromosoma codifica la expresión del árbol utilizando cualquier recorrido sobre él. Para poder emplear esta representación se hace necesario definir un conjunto de símbolos de la gramática: a) variables y constantes del problema: $T = \{x_1, x_2\}$ y b) Funciones/instrucciones del programa: $F = \{+, -, *, /, while, if-then-else\}$.

La población inicial de programas (árboles de expresiones) ha de constar de una variedad de estructuras. Para ello, se realiza una generación aleatoria de los distintos árboles de expresiones, aplicando aleatoriamente y de forma progresiva las reglas de producción de la gramática. El tamaño de los árboles ha de ser menor que un máximo permitido.

En cuanto a la función de adaptación, esta debe especificar una medida de la calidad del programa codificado en la resolución del problema. Para poder evaluar los árboles de expresiones es necesario realizar una evaluación recursiva. Una vez evaluados, se eligen dos árboles padres mediante un esquema de selección para realizar el cruce. Para ello, en primer lugar se ha de seleccionar de forma aleatoria un subárbol de cada padre. Posteriormente, los subárboles son intercambiados, generando así dos nuevos descendientes. Con el operador de cruce, los árboles pueden crecer de forma indefinida si no se limita el tamaño en la operación. Si el subárbol escogido en el segundo padre provoca que el descendiente supere el tamaño máximo, lo habitual es no realizar el cruce.

El operador de mutación puede producir una mayor o menor alteración en el descendiente, dependiendo si se realiza una mutación en un punto (dado un nodo, se cambia su valor por otro del mismo tipo) o una mutación por un subárbol aleatorio (se substituye un subárbol dado por otro generado de forma aleatoria).

Seguidamente, el resto del capítulo se centra en la descripción de los AGs.

2.3. Definición

Los AGs son algoritmos de búsqueda, de propósito general, basados en la evolución natural y la genética, que proporcionan un mecanismo de búsqueda y optimización robusto en sistemas complejos, y ofrecen una vía útil para problemas de optimización que requieran eficiencia y eficacia Goldberg [1989], Michalewicz [1992].

Los AGs usan una analogía directa con el comportamiento natural. Operan sobre una población de individuos o cromosomas, representando posibles soluciones al problema que se desea resolver, que evolucionan de generación en generación a través de un proceso de competición. A cada individuo se le asigna una puntuación, que indica el grado de adecuación de dicha solución. Mediante un mecanismo de selección se escogen los individuos para reproducirse. Cuanto mayor sea la adaptación de un individuo al problema, mayor será la probabilidad de que el mismo sea seleccionado para reproducirse, cruzando su material genético con otro individuo. Por el contrario, cuanto menor sea la adaptación de un individuo, menor será su probabilidad de ser seleccionado para la reproducción y, por tanto, de que su material genético se propague en sucesivas generaciones. Como resultado de la aplicación de los operadores genéticos de cruce y mutación sobre la población, esta evoluciona hacia una nueva generación donde los individuos mejor adaptados sustituyen a aquellos peor adaptados. Estos operadores genéticos de cruce y mutación tienen asociados las probabilidades de cruce, P_c, y de mutación, P_m, respectivamente. La P_c es la probabilidad de aplicar el operador de cruce sobre los individuos seleccionados como padres. En el caso de que este operador de cruce no se aplicase, la descendencia se obtendría mediante la duplicación de los padres. La P_m es la probabilidad de aplicar el operador de mutación sobre cada hijo de forma individual, alterando de forma aleatoria cada gen del mismo.

La figura 2.4 muestra el ciclo evolutivo de un AG. Si el AG ha sido bien diseñado, la población convergerá hacia una solución óptima del problema. Los AGs se aplican fundamentalmente en problemas de búsqueda y optimización, debido a su capacidad para explotar la información acumulada sobre un espacio de búsqueda.

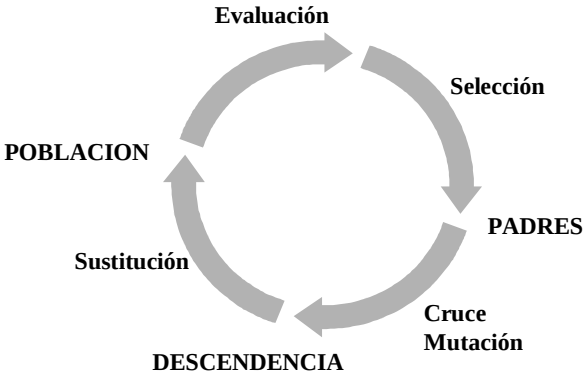

Figura 2.4: *Ciclo evolutivo en un AG.*

2.4. Modelos. Estructura

Existen dos modelos que muestran las ideas básicas para la implementación de un AG:

- *Modelo generacional (o canónico).* La estructura del AG en este modelo se muestra en la figura 2.5. Como puede apreciarse, se parte de una población inicial de cromosomas que, en la mayoría de los casos, se genera de forma aleatoria. En cada iteración, la población evoluciona mediante las siguientes operaciones:

  - evaluación de los individuos, mediante una función de evaluación especifica para cada problema.
  - selección de un conjunto de individuos (padres) mediante un mecanismo de selección.
  - aplicación de los operadores genéticos de cruce y mutación, obteniendo la descendencia.
  - sustitución de toda la población actual por los nuevos individuos, formando una nueva población.

  El proceso se repite hasta que se cumplan los criterios de finalización establecidos. En cada iteración, el AG crea una población completa con los nuevos individuos. Esta nueva población reemplaza directamente a la antigua, aunque

existe la posibilidad de preservar al mejor individuo obtenido en cada generación, teniendo así una configuración elitista. En la figura 2.6 se muestra el esquema de funcionamiento en este modelo.

Inicio

- t=0;
- Inicializar P(t)
- Evaluar P(t)
- **Mientras** (no se cumplan los criterios de finalización)

  t=t+1;

  Formar P(t) a partir de P(t-1)

  Aplicar operadores genéticos de cruce y mutación sobre P(t)

  Evaluar P(t)

Final

Figura 2.5: *Estructura de un AG generacional.*

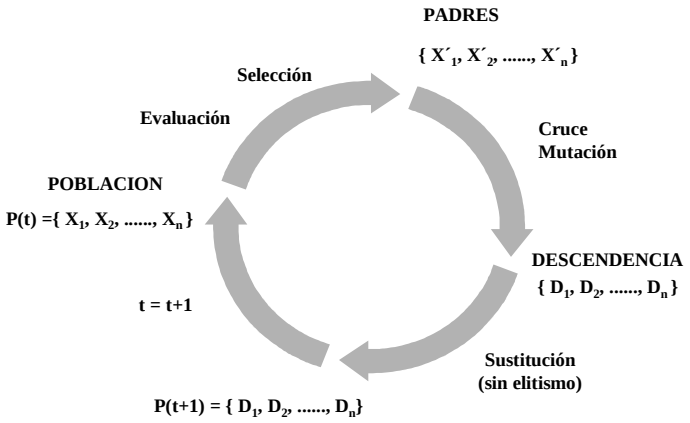

Figura 2.6: *Esquema de funcionamiento de un AG generacional.*

- *Modelo estacionario (steady-state).* En este caso, en cada iteración se reemplaza solamente una parte de la población, por lo que no sólo se deben seleccionar los individuos a ser padres sino también los individuos de la población que serán reemplazados por los descendientes, para así mantener el tamaño de

dicha población. Por tanto, en este modelo, además de las probabilidades P_c y P_m, se introduce una nueva probabilidad, denominada probabilidad de reemplazamiento generacional o de selección, P_s, que indica la proporción de la población que se reemplaza en cada iteración. Normalmente, los descendientes reemplazan a los peores individuos de la población anterior. En este caso, el modelo sería elitista ya que se preservaría al mejor individuo obtenido en cada generación. Al igual que en el algoritmo generacional, se parte de una población inicial que evoluciona en cada iteración realizando las siguientes operaciones:

- evaluación de los individuos.
- selección de un conjunto de individuos (padres).
- aplicación de los operadores genéticos de cruce y mutación, obteniendo la descendencia.
- evaluación de la descendencia.
- selección de los individuos de la población que pueden ser sustituidos por los hijos.
- decisión sobre si se sustituyen estos individuos por los nuevos, formando una nueva población.

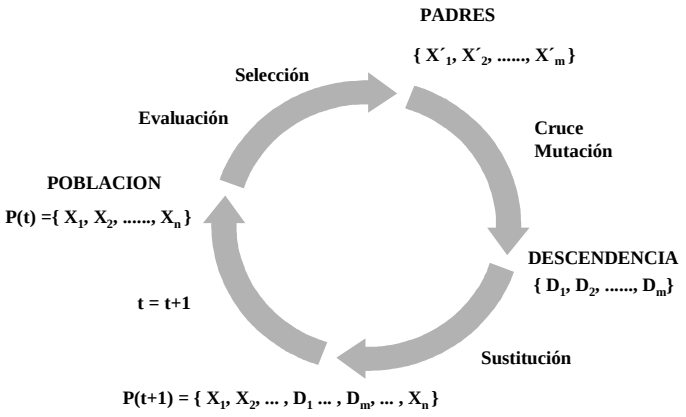

Figura 2.7: *Esquema de funcionamiento de un AG estacionario.*

En la figura 2.7 se muestra el esquema de funcionamiento en este modelo. Se ha de indicar que en la bibliografía consultada algunos autores denominan como AG estacionario a aquel en el que en cada iteración solo se reemplaza a uno o dos

individuos de la población; es decir, se seleccionan dos padres de la población, se les aplican los operadores genéticos y el ó los descendiente/s obtenidos reemplazan unicamente a uno ó dos individuos de la población [Herrera, 2004]. Sin embargo, a este procedimiento otros autores lo denominan AG incremental [Cordón et al., 2001].

Como se ha indicado anteriormente, estos dos modelos recogen las ideas básicas para la implementación de un AG; sin embargo, posteriormente, se han desarrollado otros modelos que permiten el uso de AGs en problemas donde se requiere la obtención de múltiples soluciones a un mismo problema, o la evaluación de múltiples objetivos, o donde se desea utilizar un modelo eficiente, etc. (AG CHC [Eshelman, 1991], AG con nichos [Perez et al., 2003], AG multiobjetivo [Deb., 2001, Coello et al., 2002, Konak et al., 2006, Deb, 2007], AG distribuidos [Alba y Tomassini, 2002, Nojima y Ishibuchi, 2008])

2.5. Componentes

A la hora de implementar un AG se han de definir los siguientes componentes:

- una representación genética de las soluciones para el problema dado.
- un mecanismo para crear la población inicial de soluciones.
- una función de evaluación, que permita evaluar las posibles soluciones.
- un mecanismo de selección de los individuos como padres.
- un conjunto de operadores genéticos, que alteren la composición genética de los nuevos individuos durante la reproducción.
- los valores de los parámetros que utiliza, fundamentalmente el tamaño de la población, N, la probabilidad de selección (P_s), probabilidad de cruce (P_c) y la probabilidad de mutación (P_m).
- un mecanismo de sustitución, que reemplace parte de la población actual por los nuevos individuos. Este mecanismo únicamente será necesario en el caso de un AG estacionario.

Los siguientes apartados describen cada uno de estos componentes.

2.5.1. Representación

La representación de los individuos en un AG es dependiente del problema particular a resolver y puede realizarse mediante una codificación binaria o mediante una codificación real. En la primera, cada individuo de la población se representa mediante una cadena binaria de longitud fija que codifica los valores de las variables del problema, tratándose por tanto de una codificación discreta. En este caso, el fenotipo puede ser un número entero, real, etc. En la figura 2.8 se muestra un ejemplo de representación de un individuo mediante codificación binaria natural, donde el fenotipo es un número entero.

Figura 2.8: *Representación mediante codificación binaria con fenotipo número entero.*

Es importante que individuos codificados con fenotipos cercanos tengan genotipos con cortas distancias. Esto no se cumple si se utiliza una codificación binaria natural, donde por ejemplo, los genotipos 0111 y 1000 tienen una distancia Hamming elevada, pero su significado, el fenotipo, es muy cercano, representando los valores enteros 7 y 8. Una alternativa es la utilización de código Gray en la codificación binaria, que garantiza que los valores enteros adyacentes estén representados por cadenas de bits que difieren en un simple bit. En la figura 2.9 se muestra otro ejemplo de representación de un individuo mediante codificación binaria, donde el fenotipo es un número real (en este caso números reales comprendidos entre 0.25 y 5.25).

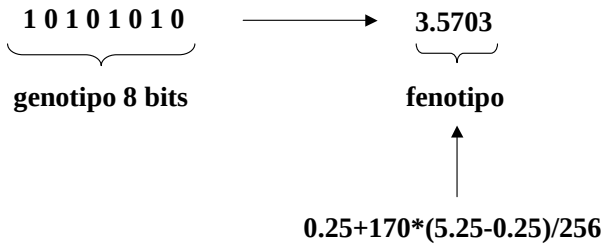

Figura 2.9: *Representación mediante codificación binaria con fenotipo número real.*

En la codificación real, cada individuo es un vector de numeros reales $X = \{x_1, ..., x_n\}, x_i \in R$. El tamaño del individuo se mantiene igual a la longitud del vector solución del problema y los valores de los genes que componen el individuo deben permanecer en el intervalo establecido para la variable que representa. Este tipo de codificación está indicado para problemas de optimización de parámetros con variables sobre dominios continuos.

2.5.2. Métodos para crear la población inicial

Normalmente, la población inicial se genera aleatoriamente, de forma uniforme, creando individuos al azar dentro del intervalo de variación que corresponda en cada problema. Si es una codificación binaria, cada gen de cada individuo será 0 ó 1 con una probabilidad del 50 %. Si se trata de una codificación real, los valores serán elegidos al azar dentro del intervalo de variación dado a cada variable.

También existe la posibilidad de generar la población inicial a partir de alguna técnica heurística o de optimización local previa. Esto puede acelerar la convergencia del AG pero, a su vez, puede ser la causa de una convergencia prematura.

2.5.3. Función de evaluación

Un aspecto importante en el comportamiento del AG es la evaluación de los individuos, que permite obtener una medida del grado de adaptación del individuo al entorno. La evaluación indica si los individuos de la población representan, o no, buenas soluciones al problema a resolver, por lo que la función debe ser diseñada para cada problema de forma específica, al igual que ocurría con la representación de los individuos.

En muchos casos se utilizan indistintamente los términos de *evaluación* y de *aptitud (fitness)*; sin embargo, es importante distinguir entre la función de evaluación y la función de aptitud utilizadas por un AG. La función de evaluación va a proporcionar, como se ha indicado anteriormente, una medida del grado de adaptación del individuo al entorno. Sin embargo, la función de aptitud va a utilizar esa medida para obtener, y asignar, la probabilidad de reproducción de dicho individuo; indicando así cómo de bueno es, comparado con los demás individuos de la población.

Por tanto, la evaluación de un individuo es independiente de la evaluación de los otros individuos, sin embargo, su aptitud viene definida con respecto al resto de los individuos de la población. Cuanto más alto es el valor de la *aptitud* o *fitness* de un individuo, mayor es su probabilidad de reproducción. Por ejemplo, si un individuo de la población, x_1, es evaluado a través de una función de evaluación f, $f(x_1)$ indicará una medida de su nivel de adaptación y es utilizada para obtener su probabilidad de reproducción (dependiente del mecanismo de selección). Si se aplica la denominada selección proporcional, cada individuo tiene una probabilidad de ser seleccionado como padre que es proporcional al valor de su función de evaluación, por lo que se tiene que $P_r(x_1) = \frac{f(x_1)}{\sum_{j=1}^{N} f(x_j)}$, siendo N el tamaño de la población.

La evaluación de los individuos de una población constituye el paso más costoso, desde el punto de vista computacional, del proceso de un AG. Una técnica que se ha venido utilizando, en el caso en que la computación de la función de evaluación sea muy compleja, es la denominada evaluación aproximada de la función. En algunos casos, la obtención de n funciones aproximadas puede resultar mejor que la evaluación exacta de una única función de evaluación (supuesto el caso de que la evaluación aproximada resulta como mínimo n veces más rápida que la evaluación exacta). Otra alternativa para mejorar la evolución de los AGs es la utilización de AGs distribuidos o paralelos (AGD), puesto que un AG es una búsqueda aleatoria en paralelo con un control centralizado que es la selección, para la que se necesita el promedio de la población[Herrera et al., 1995a]. Este resultado debe ser sincronizado, por lo que es dificil realizar una implementación eficiente sobre procesadores paralelos. Una aproximación a los AGD es el modelo de islas, donde la población de individuos se divide en subpoblaciones que evolucionan independientemente. Cada subpoblación se asigna a un procesador y el AG opera sobre dicha subpoblación. Ocasionalmente se producen migraciones entre ellas, permitiéndoles intercambiar material genético. Con la utilización de la migración, este modelo puede explotar las diferencias existentes en las subpoblaciones, representando una fuente de diversidad genética. Sin embargo, si un gran número de individuos emigran en cada generación puede ocurrir una mezcla global, eliminándose las diferencias locales. Dado que cada subpopulación es una "isla", se debe definir un procedimiento por medio del cual se mueva el material genético de una a otra. El proceso de comunicación que requiere este modelo es relativamente bajo debido a que únicamente existe migración de un número relativamente pequeño de individuos. Se pueden distinguir diferentes modelos de islas en función de la comunicación entre las subpoblaciones. Algunas

comunicaciones típicas son las siguientes:

- *Comunicación en estrella*, en la cual existe una subpoblación que es seleccionada como maestra (aquella que tiene mejor media en el valor de la función de evaluación) siendo las demás consideradas como esclavas. Todas las subpoblaciones esclavas mandan sus N mejores individuos a la subpoblación maestra, la cual a su vez manda sus M mejores individuos a cada una de las subpoblaciones esclavas.

- *Comunicación en red*, en la que no existe una jerarquía entre las subpoblaciones mandando todas y cada una de ellas sus N mejores individuos al resto de las subpoblaciones.

- *Comunicación en anillo*, en la cual cada subpoblación envía sus N mejores individuos a una población vecina, efectuándose la migración en un único sentido de flujo.

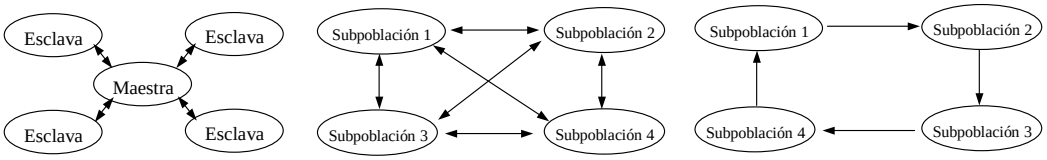

Figura 2.10: *Algoritmos genéticos distribuidos con representación en estrella, red y anillo.*

Por otro lado, existen ciertos casos donde es recomendable efectuar modificaciones o transformaciones en la función de evaluación durante la ejecución del AG. Por ejemplo, si la selección de individuos se realiza de forma proporcional al valor de su función de evaluación puede darse que la convergencia del AG sea muy rápida (convergencia prematura) y que el algoritmo converja hacia óptimos locales. Esto es debido a que al inicio de la ejecución del AG pueden existir individuos con una adaptación muy superior al resto y a medida que avanza dicha ejecución dominan a la población. Para evitarlo, se puede realizar una transformación de la función de evaluación de forma que en los inicios de la ejecución del AG se atenúe la acción de esos "superindividuos" para que no dominen rápidamente al resto de la población.

2.5.4. Métodos de selección

El mecanismo de selección es el encargado de seleccionar los individuos que van a disponer de oportunidades de reproducirse, es decir, los individuos que van a actuar como padres. El mecanismo debe garantizar que los mejores individuos, los más aptos, tengan un mayor número de oportunidades de reproducción frente a los menos aptos, por lo que la selección de un individuo está directamente relacionado con su valor de aptitud. Sin embargo, no se debe eliminar por completo las opciones de reproducción de los individuos menos aptos ya que pueden incluir material genético útil en el proceso de reproducción. Esto es lo que define la presión selectiva, la cual determina en qué grado la reproducción está dirigida por los mejores individuos. Si se opta por un método con una alta presión selectiva la búsqueda se centra en las soluciones próximas a las mejores soluciones encontradas en ese momento. Por el contrario, si se opta por una presión selectiva menor se posibilita la exploración de nuevas regiones del espacio de búsqueda.

Si se considera una población de individuos, $P(t) = \{x_1^t, x_2^t, ..., x_N^t\}$, el mecanismo de selección produce una población intermedia, $P'(t)$, con copias de los cromosomas de $P(t)$. El número de copias recibidas de cada individuo depende de su aptitud; individuos con mayor aptitud tendrán, por lo general, una mayor contribución a $P'(t)$. El mecanismo de selección se realiza en dos pasos: 1) cálculo de la probabilidad de reproducción y, 2) muestreo de individuos a través de un determinado algoritmo.

Cálculo de la probabilidad de reproducción.

Para cada cromosoma x_i^t de la población $P(t)$ se calcula la probabilidad, $P_r(x_i^t)$, de incluir una copia suya en la población $P'(t)$. Algunos métodos de cálculo de probabilidades de reproducción son:

- Proporcional Selection -Selección Proporcional- [Holland, 1975, Goldberg, 1989]. En este método se asigna una probabilidad de reproducción proporcional al valor de la adaptación del individuo, donde $P_r(x_i^t)$, i=1,..., N, se calcula como:

$$P_r(x_i^t) = \frac{f(x_i^t)}{\sum_{j=1}^{N} f(x_j^t)}$$

En la tabla 2.1 se muestra un ejemplo para una población de 10 individuos,

donde se indica una evaluación para cada individuo, dada por una función de evaluación y la probabilidad de reproducción asociada a cada uno.

Tabla 2.1: *Probabilidades de reproducción mediante Proporcional Selection.*

Individuo	$f(x_i)$	$P_r(x_i)$
1	1.2	0.11
2	0.2	0.02
3	0.6	0.06
4	1.8	0.16
5	1.0	0.09
6	2.0	0.18
7	1.4	0.13
8	0.4	0.03
9	1.6	0.15
10	0.8	0.07

- Linear Ranking Selection -Orden Lineal- [Baker, 1987a]. En este mecanismo los individuos de la población son ordenados en función de su evaluación y se asigna una probabilidad de reproducción a cada uno dependiendo de su orden, *rank*(x_i^t), siendo *rank*(x_{mejor}^t)= 1. La probabilidad de reproducción de cada individuo se calcula como:

$$P_r(x_i^t) = \frac{1}{N}(\eta_{max} - (\eta_{max} - \eta_{min})\frac{rank(x_i^t)-1}{N-1})$$

donde $\eta_{min} \in [0, 1]$ especifica el número de selecciones esperadas del peor individuo (el mejor individuo tiene $\eta_{max} = 2 - \eta_{min}$). La presión selectiva viene determinada por η_{min}; si η_{min} es bajo, la presión selectiva es alta, mientras que si es alto, la presión es baja. En la tabla 2.2 se muestra la misma población que en la tabla 2.1 pero con las probabilidades de reproducción calculadas con este procedimiento para un η_{min}=0.5.

Este mecanismo es normalmente utilizado con el algoritmo de muestro Stochastic Universal Sampling -Muestreo Universal Estocástico- [Baker, 1987b].

Tabla 2.2: *Probabilidades de reproducción mediante Linear Ranking Selection.*

Individuo	$f(x_i)$	$P_r(x_i)$
6	2.0	0.150
4	1.8	0.138
9	1.6	0.127
7	1.4	0.116
1	1.2	0.105
5	1.0	0.094
10	0.8	0.083
3	0.6	0.072
8	0.4	0.061
2	0.2	0.050

Algoritmos de muestreo.

El algoritmo de muestreo reproduce, basándose en las probabilidades de reproducción, las copias de los cromosomas para formar la población intermedia $P'(t)$. A continuación se muestran algunos algoritmos de muestreo.

- Stochastic Sampling with Replacement -Muestreo Estocástico con Reemplazamiento- [Holland, 1975, Goldberg, 1989]. Con este algoritmo la posibilidad de ser elegido es proporcional a la probabilidad de reproducción de cada individuo, por lo que los mejores individuos son los que tienen mayores posibilidades de ser elegidos. De forma intuitiva, el mecanismo situa a la población sobre una ruleta, en la que cada una de las porciones representa a un individuo. El tamaño de cada porción corresponde a su $P_r(x_i^t)$. Los mejores individuos recibirán una mayor porción de la ruleta que los peores. Para seleccionar un individuo, se genera un número aleatorio en el intervalo [0,1] y se elige al individuo situado en esa posición de la ruleta. Esta posición se puede obtener recorriendo los individuos de la población y acumulando sus proporciones de ruleta hasta que la suma exceda el valor del número aleatorio. Por ejemplo, a partir de los datos mostrados en la tabla 2.1, si se desean seleccionar como padres 6 individuos, se generarán 6 números aleatorios en el intervalo [0,1]. Si estos números son, por ejemplo, 0.44, 0.65, 0.21, 0.32, 0.06, 0.96, los individuos seleccionados como padres serán: 9, 1, 4, 4, 6 y 8.

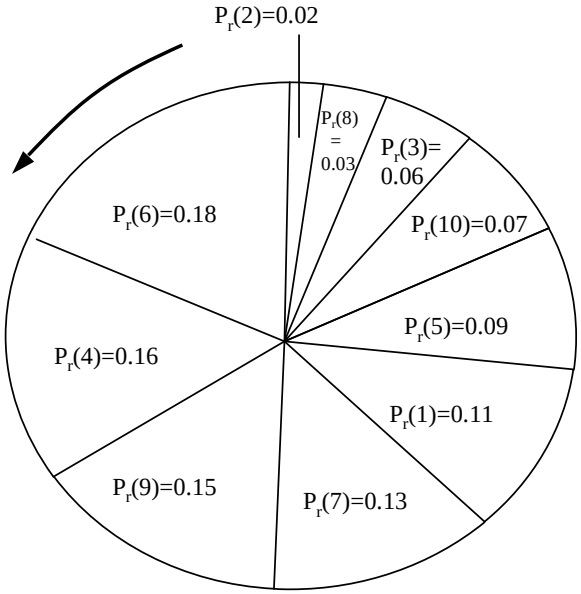

Figura 2.11: *Algoritmo de muestreo Stochastic Sampling with Replacement.*

- Stochastic Universal Sampling -Muestreo Universal Estocástico- [Baker, 1987b]. Este algoritmo garantiza que el número de copias de cualquier individuo esté acotado por el número entero inferior y superior del número de copias esperadas, $P_r(x_i^t) * N$. De forma intuitiva, al igual que en el algoritmo de la ruleta, se sitúa a la población sobre una ruleta en la que cada una de las porciones representa a un individuo. El tamaño de cada porción corresponde a su probabilidad de reproducción. Para seleccionar los individuos, se genera un número aleatorio z en el intervalo $[0, \frac{1}{r}]$, siendo r el número de individuos a seleccionar como padres, y se selecciona el individuo en cuya porción ha caido. Posteriormente, y mientras r sea menor que 1, se suma $\frac{1}{r}$ a z, seleccionándose el individuo correspondiente.

  Por ejemplo, a partir de los datos mostrados en la tabla 2.2, si se desean seleccionar como padres a 6 individuos, se generará un número aleatorio en el intervalo $[0,\frac{1}{6}]$. Si r = 0.1, los individuos seleccionados como padres serán: 6, 4, 7, 1, 10 y 8, como se muestra en la figura 2.12.

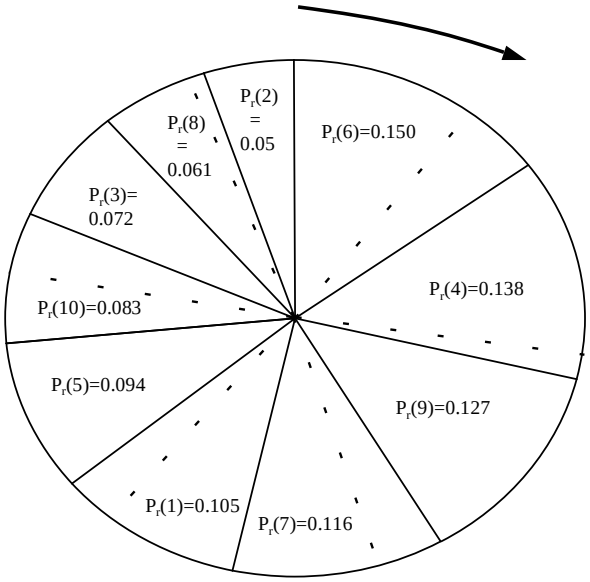

Figura 2.12: *Algoritmo de muestreo Stochastic Universal Sampling.*

- Tournament Sampling -Torneo-. En este algoritmo, las copias de los individuos se realizan en base a comparaciones directas entre ellos. Se eligen aleatoriamente k individuos de la población y se selecciona al mejor de ellos. Al valor de k se le denomina tamaño del torneo e incide directamente sobre la presión selectiva. Cuantos más individuos participan en cada torneo, la presión selectiva es mayor y los peores individuos apenas tienen oportunidades de reproducción. Por el contrario, cuando el tamaño del torneo es reducido, la presión de selectiva disminuye y los peores individuos tienen más oportunidades de ser seleccionados. En la figura 2.13 se muestra un ejemplo de este algoritmo a partir de la población mostrada en la tabla 2.1.

Un mecanismo de selección puede incorporar una estrategia elitista si se preserva al mejor o mejores individuos obtenidos en cada una de las generaciones.

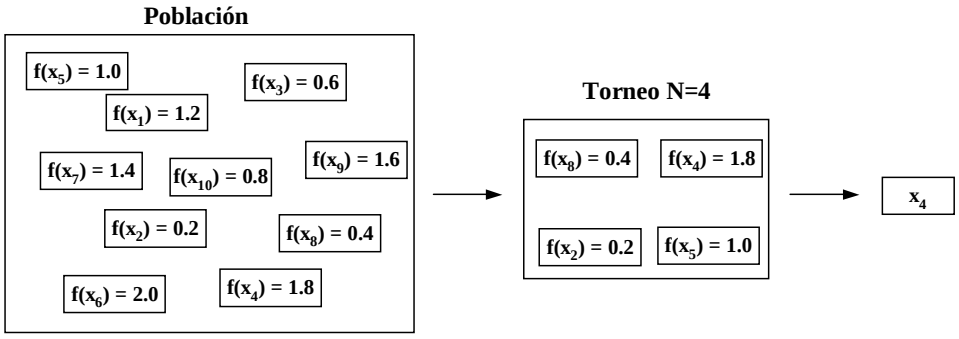

Figura 2.13: *Algoritmo Tournament Sampling.*

2.5.5. Operadores genéticos. Codificación binaria y real

Los operadores genéticos están basados en el cruce y la mutación de la información genética de los individuos seleccionados como padres de la población. El operador de cruce se considera como el más importante de los dos operadores y permite mezclar la información genética de los padres con objeto de obtener la nueva descendencia. Así, cada hijo hereda características de cada uno de sus padres. Si al realizar el cruce, el nuevo descendiente agrupa las buenas caracteristicas de los padres, podrá tener una mejor bondad que sus progenitores. Si por el contrario, el cruce no agrupa las mejores caracteristicas, es posible que el hijo tenga una peor bondad que los padres. El objetivo del cruce es explorar las regiones del espacio de búsqueda de una manera más eficiente que una búsqueda al azar, produciendo individuos válidos. El operador de cruce debe ser utilizado, en principio, con una probabilidad de actuación alta sobre los padres seleccionados (tradicionalmente se utilizan probabilidades entre un 60 % y un 95 %). Cuando se produce el cruce, se dice que la reproducción es de tipo sexual. Si operador de cruce no actúa, serán los mismos padres los descendientes del proceso de recombinación, insertándose en la nueva población. En este caso se dice que la reproducción es de tipo asexual.

Por otro lado, una vez generados los nuevos individuos, tras aplicar el operador de cruce, el operador de mutación puede realizar alteraciones aleatorias sobre la información genética de dichos individuos. En principio, debe ser aplicado con una probabilidad de actuación muy baja (tradicionalmente se utilizan probabilidades entre un 0.1 % y 1 %) ya que de lo contrario, el AG degeneraría en una búsqueda aleatoria. Los descendientes obtenidos mediante una reproducción asexual también pueden ser afectados por el este operador.

Ambos operadores, tanto el de cruce como el de mutación, deben ser diseñados teniendo en cuenta el tipo de representación utilizado en el AG. A continuación, se muestran los principales operadores de cruce[4] y mutación para una representación binaria y real de los individuos.

Operadores de cruce. Representación binaria.

- Single Point Crossover (SPX) -Cruce de 1 punto-. Los individuos que actúan como padres son recombinados por medio de la selección aleatoria de un punto de corte para, posteriormente, intercambiar los dos segmentos que se encuentran a la derecha de dicho punto. La figura 2.14 muestra el funcionamiento de este operador. También se han realizado investigaciones sobre el comportamiento de un operador de cruce basado en múltiples puntos. En el estudio realizado por De Jong [DeJong, 1975] se concluye que un cruce basado en dos puntos representa una mejora sobre el de un punto.

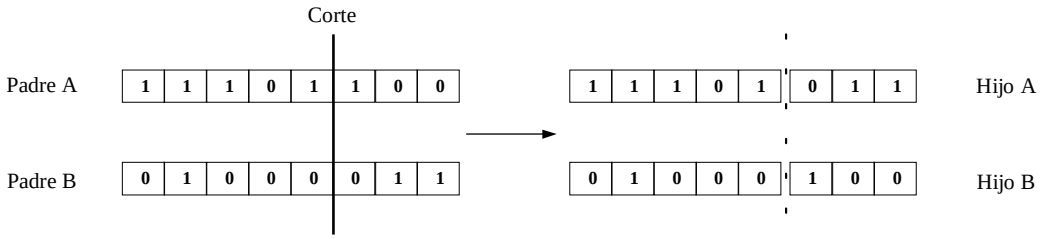

Figura 2.14: *Operador de cruce de un punto para representación binaria.*

- Double Point Crossover (DPX) -Cruce de 2 puntos-. En este caso, en lugar de cortar por un único punto los individuos que actuan como padres, se realizan dos cortes. Ha de tenerse en cuenta que ninguno de estos puntos de corte debe coincidir con el extremo de los individuos, para así garantizar que se originen tres segmentos. Los hijos son generados seleccionando el segmento central de uno de los padres y los segmentos laterales del otro padre, como puede apreciarse en la figura 2.15. Hay que tener en cuenta que, al tener más de un punto de cruce, el espacio de búsqueda del problema puede ser explorado

[4]Únicamente se muestran los operadores de cruce que se aplican sobre un dos individuos, generando dos descendientes. Existen otros operadores de cruce multi-padres y multiple-descendientes.

más a fondo; sin embargo, ello también provoca que aumente la probabilidad de romper la estructura de los buenos individuos.

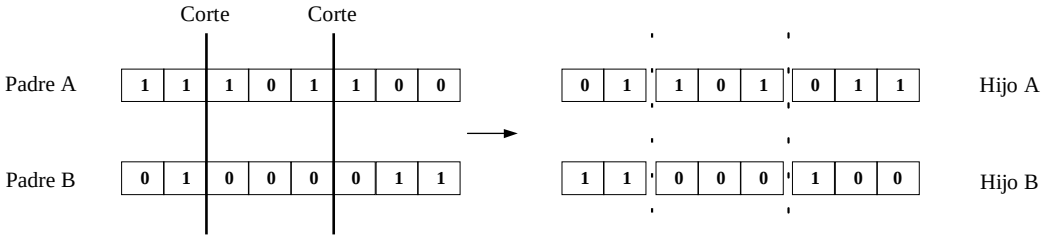

Figura 2.15: *Operador de cruce de dos puntos para representación binaria .*

- Uniform Point Crossover (UPX) -Cruce uniforme-. Mediante la aplicación de este operador se consigue que cada gen de la descendencia tenga las mismas probabilidades de pertenecer a uno u otro padre. El operador conlleva la generación de una máscara de cruce con valores binarios; para el primer descendiente, cuando exista un 1 en dicha máscara, el gen es copiado del primer padre, mientras que, cuando exista un 0, el gen es copiado del segundo padre. Para el segundo descendiente, o se intercambian los papeles de los padres o se intercambia la interpretación de los 1's y 0's de la máscara, como puede apreciarse en la figura 2.16. De este modo, la descendencia tiene una mezcla uniforme de genes de cada uno de los padres.

  Algunos autores, a la hora de generar la máscara de cruce, tienen en cuenta el valor de la funcion de evaluación de cada uno de los padres, de forma que cuanto mayor sea su bondad más probable sea heredar sus caracteristicas.

Operadores de cruce. Representación real.

A continuación, se muestran los principales operadores de cruce para una representación real. En todos los casos, los individuos seleccionados como padres son representados como $X_1 = \{x_1^1, x_2^1, \ldots, x_n^1\}, x_i^1 \in R$ y $X_2 = \{x_1^2, x_2^2, \ldots, x_n^2\}, x_i^2 \in R$.

- Flat Crossover -Cruce plano- [Radcliffe, 1991]. Con este operador se generan los descendientes $H_k = \{h_1^k, \ldots, h_i^k, \ldots, h_n^k\}$, $h_i^k \in R$, $k = 1, 2$, donde

$$h_i^k = \lambda_{i,k} x_i^1 + (1 - \lambda_{i,k}) x_i^2$$

  y $\lambda_{i,k}$ es un número aleatorio que pertenece al intervalo $[0, 1]$.

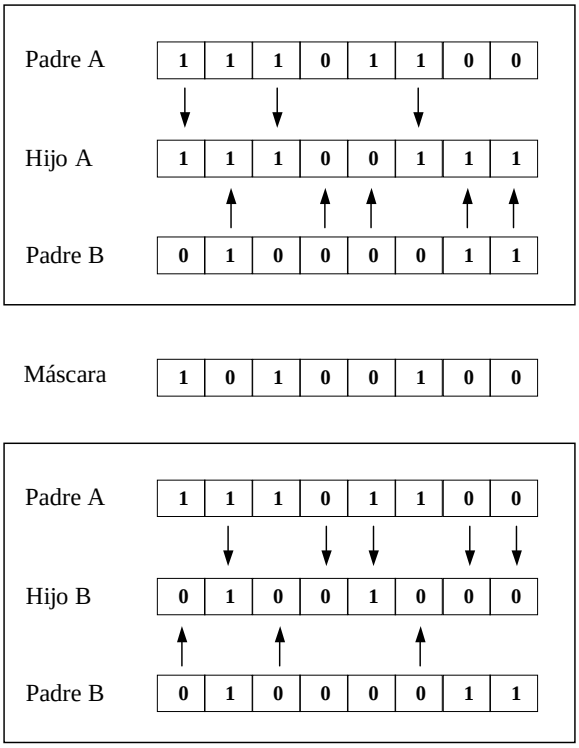

Figura 2.16: *Operador de cruce uniforme para representación binaria.*

- Lineal Crossover -Cruce lineal- [Wright, 1991]. En este caso se generan tres descendientes $H_k = \{h_1^k, h_2^k, \ldots, h_n^k\}$, $k = 1, 2, 3$, donde:

$$h_i^1 = \frac{1}{2}x_i^1 + \frac{1}{2}x_i^2$$
$$h_i^2 = \frac{3}{2}x_i^1 - \frac{1}{2}x_i^2$$
$$h_i^3 = -\frac{1}{2}x_i^1 + \frac{3}{2}x_i^2$$

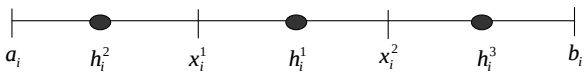

Figura 2.17: *Operador de cruce lineal para representación real.*

Los dos descendientes que obtengan un mayor valor de adaptación conformarán la descendencia.

- Simple Crossover -Cruce simple- [Michalewicz, 1992, Wright, 1991]. Este tipo de cruce es análogo al cruce de un punto utilizado en la representación binaria. De forma aleatoria, se genera una posición $k \in [1, \ldots, n-1]$ y los dos nuevos cromosomas se generan intercambiando los segmentos posteriores a esa posición, quedando los cromosomas hijos como se muestra a continuación:

$$H_1 = \{x_1^1, x_2^1, \ldots, x_k^1, x_{k+1}^2, x_n^2\}$$
$$H_2 = \{x_1^2, x_2^2, \ldots, x_k^2, x_{k+1}^1, x_n^1\}$$

- Two Point Crossover -Cruce de 2 puntos- [Eshelman et al., 1989]. Este operador es similar al cruce de dos puntos utilizado en la representación binaria. Se generan dos puntos de cruce aleatorios $i, j \in [1, n-1]$ con $i < j$. Los dos nuevos cromosomas son generados intercambiando los segmentos de los padres definidos por estos dos puntos, como se muestra a continuación.

$$H_1 = \{x_1^1, x_2^1, \ldots, x_i^2, x_{i+1}^2, \ldots, x_j^2, x_{j+1}^1, \ldots, x_n^1\}$$
$$H_2 = \{x_1^2, x_2^2, \ldots, x_i^1, x_{i+1}^1, \ldots, x_j^1, x_{j+1}^2, \ldots, x_n^2\}$$

- Uniform Crossover -Cruce uniforme- [Syswerda, 1989]. Este tipo de cruce es similar al cruce uniforme utilizado en la representación binaria. Se generan los descendientes $H_k = \{h_1^k, \ldots, h_i^k, \ldots, h_n^k\}$, $k = 1, 2$. El valor de cada gen h_i^1 es determinado por la elección, de forma aleatoria (uniformemente), de los valores x_i^1 o x_i^2 de los padres:

$$h_i^k = \begin{cases} x_i^1 & \text{si } \tau = 0 \\ x_i^2 & \text{si } \tau = 1 \end{cases}$$

donde τ es un número aleatorio que puede tener el valor de 0 ó 1.

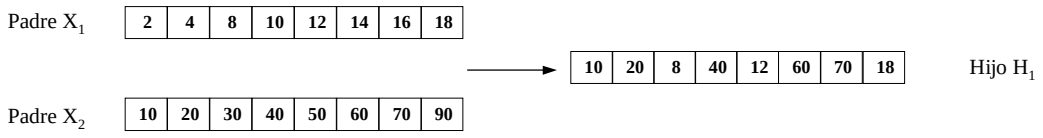

Figura 2.18: *Operador de cruce uniforme para representación real.*

- BLX-α Crossover -Cruce BLX-α- [Eshelman y Schaffer, 1993]. Este tipo de cruce es una extensión del cruce plano y permite que los genes de la descendencia puedan estar localizados fuera del intervalo $[x_i^1, x_i^2]$. Así, se generarán los descendientes $H_k = \{h_1^k, h_2^k, \ldots, h_n^k\}$, $k = 1, 2$, donde h_i^k es un valor escogido aleatoriamente en el intervalo $[X_{min} - I\alpha, X_{max} + I\alpha]$, con $X_{max} = max\{x_i^1, x_i^2\}$, $X_{min} = min\{x_i^1, x_i^2\}$, $I = X_{max} - X_{min}$, $\alpha \in [0, 1]$. Como puede apreciarse, si α =0.0 este operador es idéntico al cruce plano.

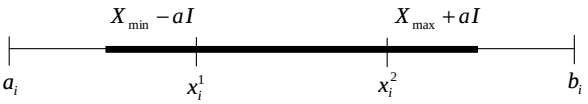

Figura 2.19: *Operador de cruce BLX-α para representación real.*

- Arithmetical Crossover -Cruce aritmético- [Michalewicz, 1992]. Este tipo de cruce se define como una combinación lineal de 2 vectores. Por tanto, se generarán los cromosomas hijos $H_1 = \{h_1^1, h_2^1, \ldots, h_n^1\}$ y $H_2 = \{h_1^2, h_2^2, \ldots, h_n^2\}$, donde:

$$h_i^1 = \lambda x_i^1 + (1 - \lambda) x_i^2; \qquad h_i^2 = \lambda x_i^2 + (1 - \lambda) x_i^1$$

El operador puede utilizar el parámetro $\lambda \in [0,1]$ como constante (cruce aritmético uniforme) o bien, como una variable que depende de la edad de la población (cruce aritmético no uniforme).

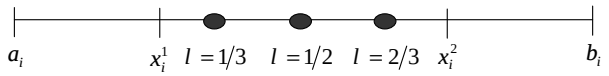

Figura 2.20: *Operador de cruce aritmético con diferentes λ para representación real.*

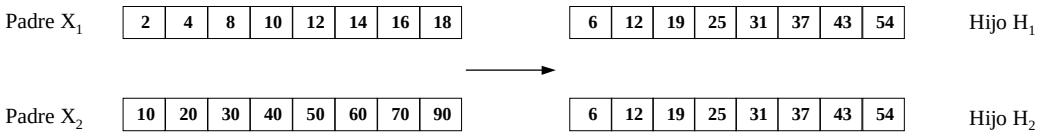

Figura 2.21: *Operador de cruce aritmético (λ=0.5) para representación real.*

- Extended Line Crossover -Cruce lineal extendido- [Mühlenbein y Schlierkamp-Voosen, 1993]. En este caso se generan los descendientes $H_k = \{h_1^k, h_2^k, \ldots, h_n^k\}$, $k = 1, 2$, donde

$$h_i^k = x_i^1 + \lambda_k(x_i^2 - x_i^1)$$

y λ_k es un valor elegido de forma aleatoria (uniformemente) en el intervalo [-0.25, 1.25].

- Extended Intermediate Crossover -Cruce intermedio extendido- [Mühlenbein y Schlierkamp-Voosen, 1993]. Se generan los descendientes $H_k = \{h_1^k, h_2^k, \ldots, h_n^k\}$, $k = 1, 2$, donde

$$h_i^k = x_i^1 + \lambda_{i,k}(x_i^2 - x_i^1)$$

y $\lambda_{i,k}$ es un valor elegido de forma aleatoria (uniformemente) en el intervalo [-0.25, 1.25]. Este operador es equivalente al BLX-0.25.

- Fuzzy Connectives Based Crossover (FCB) -Cruce basado en el uso de conectivos difusos- [Herrera et al., 1995a]. En este tipo de cruce, dados dos genes,

x_i^1 y x_i^2, que se van a cruzar, x_i^1,x_i^2 $\in [a_i, b_i]$, el intervalo de actuación del gen i $[a_i, b_i]$ se puede dividir en tres regiones para obtener los descendientes $[a_i, \alpha_i]$, $[\alpha_i, \beta_i]$, y $[\beta_i, b_i]$, siendo $\alpha_i = min(x_i^1, x_i^2)$, $\beta_i = max(x_i^1, x_i^2)$. También se pueden considerar los intervalos $[\alpha_i^{'}, \beta_i^{'}]$ con $\alpha_i^{'} \leq \alpha_i$ y $\beta_i^{'} \geq \beta_i$. Estos 3 intervalos pueden ser clasificados como intervalos de exploración o explotación. La variedad de descendientes en el intervalo de actuación garantiza una diversidad de la población y previene la convergencia prematura. Para conseguir esto, se propone la utilización de cuatro funciones F, S, M y L definidas de $[a, b]x[a, b]$ en $[a, b], a, b \in R$ y que cumplen:

a) $\forall x, y \in [a, b]$ $F(x, y) \leq min(x, y)$
b) $\forall x, y \in [a, b]$ $S(x, y) \geq max(x, y)$
c) $\forall x, y \in [a, b]$ $min(x, y) \leq M(x, y) \leq max(x, y)$
d) $\forall x, y \in [a, b]$ $F(x, y) \leq L(x, y) \leq S(x, y)$
e) F, S, M y L son monótonas, no decrecientes.

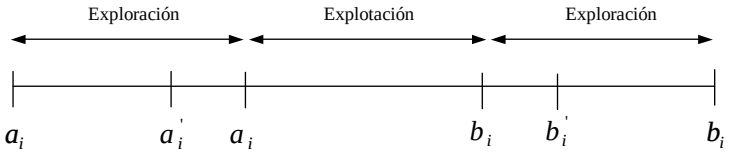

Figura 2.22: *Intervalo de actuación para un gen mediante el operador FCB*

Así, si $Q \in F, S, M, L$ y los cromosomas seleccionados como padres son X_1 y X_2, se genera un cromosoma $H = \{h_1, h_2, \ldots, h_n\}$ como $H = Q(X_1, X_2)$ donde $h_i = Q(x_i^1, x_i^2)$, $i = 1, \ldots, n$. Haciendo uso de los operadores t-normas, t-conormas, funciones promedio, y operadores de compensación generalizados que son utilizados como conectivos en lógida difusa, se asocia F a una t-norma T, S a una t-conorma G, M con un operador promedio P y L con un operador de compensación generalizado C. Para llevarlo a cabo es necesario un conjunto de transformaciones lineales para poder aplicar estos operadores en los intervalos de definición de los genes. Por tanto, utilizando estos operadores se pueden construir distintas familias de AG que se distinguen en como realizan las siguientes operaciones:

- generación de los cromosomas hijos haciendo uso de los operadores definidos.
- selección de los cromosomas hijos que formarán parte de la población.

Operadores de mutación. Representación binaria.

- Bit-flip mutation -Mutación aleatoria-. Este tipo de mutación consiste en recorrer todo el individuo y variar aleatoriamente cada gen del mismo. Así, cada bit de cada individuo de la población puede sufrir una alteración (consistente invertir el bit) de acuerdo a una cierta probabilidad, la probabilidad de mutación.

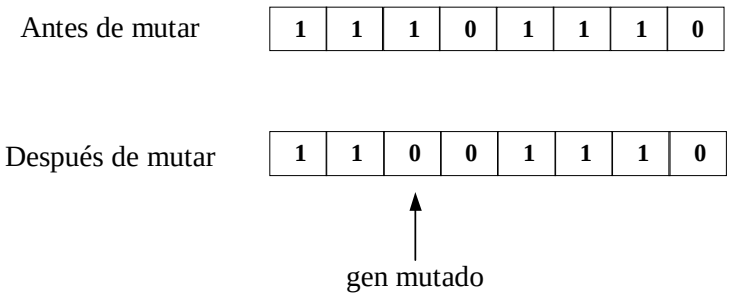

Figura 2.23: *Mutación aleatoria en representación binaria.*

- Clock mutation -Mutación de reloj- [Goldberg, 1989]. Con este operador, en lugar de recorrer todo el individuo, después de que el primer bit es mutado, la localización del siguiente bit a mutar viene determinada por una distribución exponencial de media $\mu = 1/P_m$. El procedimiento es el siguiente: en primer lugar, se genera un número aleatorio $r \in [0, 1]$. Posteriormente, se estima el siguiente bit a ser mutado saltando $\eta = -P_m Ln(1 - r)$ bits desde el actual.

- Swap mutation -Mutación de intercambio-. En este caso, la mutación se realiza intercambiando los valores de dos genes del individuo.

Operadores de mutación. Representación real.

- Random mutation -mutación aleatoria- [Radcliffe, 1991, Michalewicz, 1992]. Dado el individuo $H = \{h_1, h_2, \ldots, h_n\}$, donde h_i es el gen a ser mutado, la mutación consiste sustituir ese valor por un número generado de forma aleatoria (uniforme) en el dominio (a_i, b_i), donde a_i y b_i son los límites de los valores que h_i puede tener.

- Non-uniform mutation -mutación no uniforme- [Michalewicz, 1992]. En este caso, cuando el operador se aplica al cromosoma $H = \{h_1, h_2, \ldots, h_n\}$ en la generación t, el vector resultante será $H' = \{h_1', h_2', \ldots, h_n'\}$, donde:

$$h_i' = \begin{cases} h_i + \Delta(t, b_i - h_i) & \text{si } \tau = 0 \\ h_i - \Delta(t, h_i - a_i) & \text{si } \tau = 1 \end{cases}$$

siendo τ un número aleatorio que puede tener el valor de 0 ó 1 y,

$$\Delta(t, y) = y(1 - \lambda^{(1 - \frac{t}{t_{max}})^b})$$

donde λ es un número aleatorio en el intervalo $[0, 1]$ y t_{max} el número máximo de generaciones del AG. La función Δ devuelve un valor en el rango $[0, y]$, de forma que la probabilidad de devolver un valor cercano a cero se incrementa a medida que el algoritmo avanza. El parámetro b, definido por el usuario, determina el grado de dependencia sobre el número de generaciones del algoritmo.

- Real Number Creepp -mutación utilizando desplazamiento- [Davis, 1991]. En este tipo de mutación, los genes a ser mutados son modificados en un valor que los aumentan o disminuyen en una pequeña cantidad aleatoria. El máximo deslizamiento permitido viene determinado por un parámetro definido por el usuario.

- Mühlenbein's mutation -mutación de Mühlenbein- [Mühlenbein y Schlierkamp-Voosen, 1993]. Dado el individuo $H = \{h_1, h_2, \ldots, h_n\}$, donde h_i es el gen a ser mutado, el vector resultante al aplicar el operador será $H' = \{h_1', h_2', \ldots, h_n'\}$, siendo:

$$h_i' = h_i \pm rang_i \gamma$$

donde $rang_i$ define el rango de mutación, ajustándose normalmente a $0.1(b_i - a_i)$ (a_i y b_i son los límites de los valores que h_i puede tener). El signo + o - se elige con una probabilidad del 50 % y

$$\gamma = \sum_{k=0}^{15} \alpha_k 2^{-k}$$

donde $\alpha \in [0, 1]$ se genera de forma aleatoria con $p(\alpha_i = 1) = \frac{1}{16}$.

2.5.6. Valores de los parámetros de control

Un aspecto muy importante dentro de la estructura de un AG son los parámetros que utiliza, fundamentalmente, el tamaño de la población, N, la probabilidad de selección, P_s, la probabilidad de cruce, P_c, y la probabilidad de mutación, P_m. La elección de estos parámetros determinará, en gran medida, que el algoritmo encuentre una solución óptima, y de forma eficiente, al problema dado. Además, para obtener buenas soluciones, el AG debe establecer un balance adecuado entre:

- la exploración hacia nuevas zonas de búsqueda que puedan presentar buenos resultados y que evite el problema de la convergencia prematura, es decir, la cantidad de esfuerzo empleado en la búsqueda en regiones distantes del espacio.

- la explotación de la información que posee el algoritmo, es decir, la cantidad de esfuerzo empleado en la búsqueda en la región actual.

El encontrar un equilibrio en la relación exploración/explotación es fundamental, por un lado, para hallar rápidamente regiones del espacio con buenas soluciones y, por otro, para no gastar excesivo tiempo en buscar en regiones del espacio no prometedoras o ya explotadas.

La búsqueda de los valores de los parámetros que optimizan el comportamiento de un AG ha sido objeto de diferentes estudios desde los primeros trabajos, y sigue teniendo completa actualidad: [DeJong, 1975, Grefenstette, 1986, Schaffer et al., 1989, Bramlette, 1991, Srinivas y Patnaik, 1994, Wu y Chow, 1995, Eiben et al., 1999, Cao y Wu, 1999, Cicirello y Smith, 2000, Herrera y Lozano, 2003, Subbu y Bonissone, 2003, Sahin y Abbate, 2004, Jansen et al., 2005, Eiben et al., 2007, Meyer-Nieberg y Beyer, 2007, Cantú-Paz, 2007, DeJong, 2007, Sun, 2007, Tsai y Chao, 2008, Vajda et al., 2008].

En cuanto al tamaño idóneo de la población, N, parece intuitivo que poblaciones pequeñas corren el riesgo de no cubrir adecuadamente el espacio de búsqueda, mientras que poblaciones de gran tamaño pueden conllevar problemas relacionados con el excesivo costo computacional. En [Goldberg, 1989], Goldberg indica que el tamaño óptimo de la población crece de forma exponencial con respecto a la longitud de los individuos, λ, representados con codificación binaria. Evidentemente, con ese resultado la aplicabilidad de los AG en problemas reales sería muy limitada, puesto que no serían competitivos con otros métodos de optimización combinatoria. Así,

Alander [Alander, 1992] sugiere que un tamaño de población comprendido entre λ y 2λ es suficiente para tener un buen comportamiento.

La probabilidad de selección, P_s, controla el porcentaje de individuos de la población que son seleccionados para aplicarles los operadores genéticos. A su vez, indica el porcentaje de individuos de la población que serán reemplazados en cada generación. Un valor de 1.0 indica que todos los individuos son seleccionados y, por tanto, la población entera será reemplazada.

La probabilidad de cruce, P_c, controla la frecuencia con la que el operador de cruce se aplica. En cada nueva población, P_s*N*P_c individuos se cruzarán. A mayor P_c, más rápidamente se crearán nuevos individuos que serán introducidos en la población. Sin embargo, si P_c es demasiado elevado, se corre el riesgo que los buenos individuos sean descartados rápidamente (si no son preservados mediante alguna estrategia elitista en la selección). Por otro lado, si P_c es demasiado pequeño, la búsqueda puede estancarse debido a la baja exploración que se realiza. Como ya se ha comentado anteriormente, tradicionalmente se han utilizado valores de P_c elevados, $P_c \in$ [0.6,0.95] [Srinivas y Patnaik, 1994].

La probabilidad de mutación, P_m, controla el número de mutaciones sobre cada individuo obtenido tras aplicar el operador de cruce. Tradicionalmente se han utilizado valores de P_m pequeños, $P_m \in$ [0.001,0.01] [Herrera y Lozano, 2003]. La utilización de valores elevados de P_m podría asemejar el funcionamiento del algoritmo a una búsqueda aleatoria. Mühlenbein [Mühlenbein, 1992], a partir del trabajo realizado por Bremermann, muestra una fórmula para P_m que depende de la longitud del individuo, λ, $P_m = \frac{1}{\lambda}$. A su vez, Bäck [Bäck, 1996] también indicó que $\frac{1}{\lambda}$ podía ser un buen valor para P_m, conjuntamente a la utilización de código Gray.

Los primeros estudios en la búsqueda de los parámetros que optimizan el comportamiento de un AG fueron realizados por De Jong [DeJong, 1975] que determinó, experimentalmente, una serie de valores. En concreto, los párametros considerados por De Jong fueron N, P_c, P_m y G. Este último parámetro medía el solapamiento entre las poblaciones y su valor podía variar de 0 a 1. Si G=1, no existía solapamiento entre poblaciones, mientras que si G>0, si existía solapamiento y se seleccionaban N*G individuos para aplicarles los operadores genéticos (por tanto, el parámetro G se corresponde con la P_s comentada anteriormente). Los valores recomendados por

este autor fueron los siguientes:

- N=50, G=1.0, P_c=0.6, P_m=0.001

En estos estudios, De Jong experimentó con 5 funciones diferentes e introdujo tres tipos de medidas para evaluar el comportamiento de los AGs, que han sido utilizadas en posteriores estudios y se conocen como:

a) Evaluación on-line. En este caso, si $v_i(t)$ es la función objetivo del i-ésimo individuo ($i = 1, \ldots, N$) en la t-ésima población, la evaluación on-line después de T iteraciones, se define como:

$$v^{on-line}(T) = \frac{\sum_{t=1}^{T} \sum_{i=1}^{N} v_i(t)}{N * T}$$

por lo que mide el comportamiento medio de todos los individuos generados hasta la iteración T.

b) Evaluación off-line. Esta evaluación mide el comportamiento del AG en su proceso de convergencia hacia el óptimo, midiendo el comportamiento medio de los mejores individuos obtenidos en cada generación. Así, si $v^*(t)$ es el mejor valor de la función objetivo obtenido hasta el tiempo t, la evaluación off-line después de T generaciones, se define como:

$$v^{off-line}(T) = \frac{\sum_{t=1}^{T} v^*(t)}{T}$$

c) Evaluación basada en el mejor después de T evaluaciones, donde se evalúa el AG por medio del mejor valor de la función de evaluación encontrado en la evolución.

Posteriormente, Grefenstette [Grefenstette, 1986] propuso los siguientes parámetros para obtener el mejor comportamiento de un AG, tanto para evaluaciones on-line como off-line:

a) on-line: N=30, G=1.0, P_c=0.95, P_m=0.01

b) off-line: N=80, G=0.9, P_c=0.45, P_m=0.01

En 1989, el resultado obtenido por J.D. Schaffer [Schaffer et al., 1989] fue que, para una población comprendida entre 20 y 30 individuos, la P_m debería estar comprendida en el intervalo [0.005, 0.01] y la P_c en el intervalo [0.75, 0.95].

Como puede apreciarse, en distintos estudios, se ha tratado de encontrar un conjunto de valores óptimos y globales, es decir, un conjunto de valores que puedan ser aplicados a un gran número de problemas de optimización. Los valores indicados anteriormente estaban basados en problemas de optimización de funciones matemáticas donde la codificación utilizada en el AG fue la binaria y, probablemente, su aplicación a otros tipos de problemas esté limitada.

Por otro lado, en [Wolpert y MacReady, 1997] se muestra el teorema de *No Free Lunch -NFL-* y entre las diversas implicaciones del mismo está el hecho de que un conjunto de valores, potencialmente óptimos para un problema dado, no puede ser utilizado sobre todos los problemas de optimización y esperar que el resultado obtenido sea el óptimo, tal y como era aceptado anteriormente de forma general. Por tanto, las pruebas experimentales no proporcionarán, en general, información extrapolable a otros dominios de problemas y limita la validez de la información obtenida a través de dichos experimentos a los problemas estudiados. Cada tipo de problema necesita una configuración específica del AG para obtener un rendimiento satisfactorio y un conjunto de valores óptimos para un AG no puede ser generalizado a todos los casos, no existiendo una regla general para elegir los valores apropiados. Por ello, la elección de dichos valores representa, en muchos de esos casos, un problema de prueba y error. Así, se hacen necesarios métodos ó técnicas eficientes que ayuden a encontrar los valores óptimos de los parámetros para cada tipo de problema.

Dentro de este aspecto, la elección de los parámetros de control de un AG, se enmarca el principal objetivo perseguido en esta propuesta de tesis doctoral: el diseño, desarrollo y validación de un sistema de optimización de parámetros de un AG, a través del cual se puedan hallar unos valores óptimos para cada tipo de problema a resolver, con independencia del tipo de codificación utilizado en el AG.

2.5.7. Métodos de sustitución

Los mecanismos de sustitución seleccionan los individuos que serán eliminados de la población con objeto de ser reemplazados por los nuevos individuos generados. Como ya se ha indicado en el apartado 2.3, cuando se considera un modelo generacional (o canónico), el AG crea en cada iteración una población completa con nuevos individuos y, por tanto, se reemplaza a la población completa. Sin embargo, en el modelo estacionario (steady-state) en cada iteración se reemplaza solamente una parte de la población, por lo que se hace necesario un mecanismo de sustitución o reemplazo a través del cual los individuos de la población anterior sean reemplazados por los descendientes, teniendo en cuenta que para insertar un nuevo individuo debería de eliminarse previamente otro de la población. Evidentemente la presión selectiva viene determinada por la forma en que los individuos de la población son reemplazados por los nuevos descendientes. Existen diferentes tipos de reemplazo, que pueden ser deterministicos o aleatorios:

- Replace Random (RA) -Aleatorio-, donde el nuevo individuo se inserta en lugar de cualquiera de los individuos de la población.
- Replace Parents (RP) -Reemplazo de padres-, donde los individuos hijos de la nueva descendencia reemplazan directamente a los padres.
- Reemplazo de similares, donde una vez obtenida la evaluación de la descendencia se selecciona un grupo de individuos de la población con una evaluación similar, reemplazándose aleatoriamente los necesarios.
- Ruleta invertida, donde a cada individuo se le asigna una probabilidad de reemplazo que es inversamente proporcional a su bondad. Los individuos son reemplazados atendiendo a un algoritmo de muestreo, similar al Stochastic Sampling with Replacement pero a la inversa, ya que los individuos con mayor evaluación tienen una menor probabilidad de ser eliminados.
- Replace Worst Strategy (RW) -Reemplazo de los peores-, donde se reemplazan los peores individuos de la población, lo cual genera una alta presión selectiva.
- Restricted Tournament Selection (RTS) -Torneo restringido-, en el que se reemplaza al más parecido de entre n individuos (n=3, ...) seleccionados.
- Worst Among Most Similar Replacement (WAMS) -Reemplazar el peor entre semejantes-, donde se reemplaza el peor individuo de n padres (n=3, ...) que más se parezcan al individuo hijo generado.

- Deterministic Crowding (DC) -Algoritmo de Crowding Determinístico-, donde el individuo hijo reemplaza a su padre más parecido.

Si se utiliza una estrategia *elitista*, no se reemplaza al mejor individuo (ó a los mejores individuos) de la población, lo cual es muy aconsejado en los modelos generacionales para no perder la mejor solución encontrada. Por otro lado, si lo que se persigue es obtener un alto grado de elitismo se puede generar una población intermedia, con todos los padres y todos los descendientes, y seleccionar los mejores manteniendo el tamaño de la población.

2.6. Convergencia vs. Diversidad

Como ya ha sido comentado previamente en el apartado 2.5.6, un AG necesita establecer un equilibrio entre dos factores contrapuestos:

- la diversificación, es decir, la exploración del espacio hacia nuevas zonas de búsqueda que puedan presentar buenos resultados y,
- la intensificación, la explotación del espacio de búsqueda, es decir, la explotación de la información que posee para realizar una búsqueda en profundidad en la región actual obteniendo las mejores soluciones.

En principio, el hecho de realizar una buena exploración teniendo *diversidad* en la población puede evitar el problema de la *convergencia* prematura, en donde anticipadamente se centra la búsqueda en zonas que no contienen el óptimo global.

Por tanto, el comportamiento del AG viene determinado por esos dos factores contrapuestos: la *convergencia*, centrando la búsqueda en zonas prometedoras mediante la presión selectiva, y la *diversidad*, evitando que exista una convergencia prematura. Si todos los individuos de la población son parecidos existe falta de diversidad y ello puede conllevar a una convergencia prematura hacia óptimos locales.

> *¿Qué se puede hacer para que un AG converja manteniendo la diversidad y evitando una convergencia prematura?*

A continuación, se muestran algunas soluciones propuestas en la literatura para mejorar el comportamiento de un AG, consistentes en incluir mecanismos de diversidad en la evolución del algoritmo.

a) Diversidad en el cruce. La diversidad que introduce el operador de cruce depende:

   - de la técnica de cruce utilizada, haciendo que los padres se puedan seleccionar de forma que se mantenga la diversidad. Esto puede conseguirse de forma que un individuo no pueda cruzarse ni con sus padres, ni con sus hijos, ni con sus hermanos, ni con él mismo (prohibición de cruce basado en ascendencia). También se puede prohibir el incesto, donde dos individuos padres se cruzan si su distancia Hamming está por encima de un cierto umbral o bien, realizar un emparejamiento variado, donde un individuo se cruza con otro que es bastante diferente, analizando previamente el grado de similitud entre ellos.
   - de la técnica para generar los hijos. Dado que el operador de cruce combina las propiedades de dos o más individuos padres para generar hijos, es importante seleccionar un operador que consiga el mejor equilibrio entre exploración y explotación del espacio de búsqueda.
   - del número de padres e hijos. En este caso se pueden utilizar operadores de cruce multipadre, donde se seleccionan, para la reproducción, más de dos padres y se generan más de dos hijos. También se pueden utilizar operadores cruce con múltiples descendientes, donde a partir de dos padres se generan más de dos hijos.

b) Separación espacial. En este caso se proponen distintos métodos para evitar la convergencia prematura, siendo los más representativos los AGs celulares y los AGs distribuidos [Alba y Tomassini, 2002].

c) Estrategias de sustitución. Su principal objetivo es el de aportar una mayor convergencia o diversidad en la población en función del mecanismo de sustitución utilizado. Tienen un mayor interés en los modelos estacionarios o en el caso de existir mecanismos de competición entre padres o hijos. Existen distintos métodos basados en similitud o basados en competición, como por ejemplo:

   - Método Minimal Generation Gap (MGG), basado en competición, donde se seleccionan 2 padres de forma aleatoria, se generan λ hijos mediante el operador de cruce y los padres son reemplazados por: a) el mejor individuo de entre los padres y los hijos y, b) otro individuo obtenido mediante el método de la ruleta.

- Método Distance Dependent Alternation (DDA), basado en similitud, donde se seleccionan μ padres aleatoriamente. Posteriormente, se generan λ hijos mediante el operador de cruce y, si el mejor hijo mejora al padre más cercano, lo sustituye, si no, se busca otro padre de forma aleatoria hasta que lo mejore.

d) Meta-evolución. Esta solución consiste en buscar el mejor AG para resolver un problema como un problema de optimización y se utiliza otro AG para resolverlo, es decir, existe un meta-AG que opera sobre una población de AGs, donde cada AG es un individuo con sus componentes que intentan resolver el problema dado.

e) Adaptación de los componentes del AG. Esta solución consiste en adaptar o ajustar dinámicamente algunos componentes del AG, en definitiva, cambiar su configuración durante su ejecución, modificándolos en función de su estado o de la información que se tenga sobre el espacio de búsqueda. Cuando se produce esta adaptación se habla de Algoritmos Genéticos Adaptativos (AGAs) -Adaptive Genetic Algorithms-, que son descritos en el capítulo 3.

2.7. Conclusiones

En este segundo capítulo se ha presentado un resumen del estado actual del arte en Algoritmos Genéticos. Se han revisado los conceptos básicos de funcionamiento, los modelos básicos de implementación, sus estructuras, asi como los componentes de que constan.

La eficacia de un Algoritmo Genético depende en buena parte de la elección de sus componentes (representación, operadores, parámetros de control, etc.). La variedad de parámetros existentes en estos componentes, la posible adaptación de los mismos y la complejidad de las interacciones entre dichos componentes y parámetros, hace que la búsqueda de una configuración óptima en un Algoritmo Genético sea muy difícil, si no imposible. Si nos centramos únicamente en los parámetros de control, ¿como se puede encontrar la "mejor" configuración del algoritmo para un problema dado? Una opción es realizar un proceso de ajuste o "tuning" de dichos parámetros, intentando encontrar "buenos" valores antes de la ejecución del algoritmo. Sin embargo, esto no es muy aconsejable ya que puede llegar a ser bastante desesperante. Así, y dado que un AG es un proceso intrinsecamente dinámico, parece

más acertado introducir algún tipo de adaptación dentro del algoritmo, cuestión que es tratada en el siguiente capítulo.

Capítulo 3

Algoritmos Genéticos Adaptativos

3.1. Introducción

Los Algoritmos Genéticos Adaptativos (AGAs) ajustan dinamicamente los componentes de un AG durante su ejecución, es decir, configuran el algoritmo mientras se resuelve el problema. Su principal objetivo es ofrecer un comportamiento más apropiado, tanto en la exploración como en la explotación de la información, para evitar una convergencia prematura en el problema y así mejorar el resultado final.

El encontrar operadores genéticos robustos o un conjunto de parámetros que eviten la convergencia prematura en cualquier tipo de problema no es una tarea trivial puesto que son dependientes del problema [Bäck, 1992]. Además, diferentes operadores y diferentes valores de los parámetros pueden ser los óptimos en distintos estados del proceso evolutivo para obtener un buen equilibrio en la relación explotación/exploración. Por estas razones, algunos autores han sugerido la utilización de técnicas adaptativas, donde la configuración del AG es modificada en función de la información que se tenga sobre el espacio de búsqueda. Así, por ejemplo, en [Eiben et al., 1999, Bäck, 1991, 1992, Hesser y Männer, 1991] se argumenta que la utilización de valores fijos durante la ejecución de un AG es inapropiada, ya que se trata de un proceso intrínsecamente dinámico, adaptativo y, por tanto, el uso de parámetros fijos o estáticos que no cambian su valor en la ejecución va en contra de ese espíritu y puede conducir a un menor rendimiento del algoritmo.

3.2. Clasificación

Hasta ahora se ha comentado que un AGA ajusta dinamicamente los componentes durante la ejecución de un AG, pero a la hora de diseñarlo surgen las siguientes cuestiones:

1. *¿qué componente o elemento se cambia o se ajusta en el AG?*
2. *¿cómo se realiza el cambio? ¿qué tipo de adaptación se realiza?*
3. *¿a qué nivel se realiza el ajuste?*

Así, los Algoritmos Genéticos Adaptativos pueden clasificarse atendiendo:

1. al componente que se adapta.
   - *a*) Representación.
   - *b*) Función de evaluación.
   - *c*) Método de selección.
   - *d*) Operadores genéticos.
   - *e*) Parámetros de control.
2. al tipo de adaptación que se realiza.
   - *a*) Determinístico.
   - *b*) Adaptativo.
   - *c*) Auto-adaptativo.
3. al nivel en el que se realiza la adaptación.
   - *a*) Entorno.
   - *b*) Población.
   - *c*) Individuo.
   - *d*) Componente.

Los siguientes apartados muestran con mayor detalle esta clasificación, prestando especial atención a la adaptación de los parámetros de control.

3.2.1. Componentes de adaptación

Respondiendo a la pregunta *¿qué componente o elemento se cambia?* y, por tanto, atendiendo al elemento que se adapta, los AGAs se pueden clasificar de la siguiente forma:

1. Adaptación de la representación de los individuos, como proponen: Shaefer, con su método denominado *ARGOT* [Shaefer, 1987]; Whitley, con *Delta coding* [Whitley et al., 1991]; Schraudolph y Belew, con el método *Dynamic Parameter Encoding* [Schraudolph y Belew, 1992] o Goldberg, con *Messy GAs (mGAs)* [Goldberg et al., 1991].

2. Adaptación de la función de evaluación. En este caso se modifica la función de evaluación utilizada en el AG. Aunque sin ser la técnica de adaptación más utilizada, proporciona un buen mecanismo que incrementa el comportamiento del AG [Eiben et al., 1999]. Por ejemplo, la función de evaluación podría atenuar, durante las primeras iteraciones del AG, la acción de aquellos individuos que tienen un valor de evaluación muy por encima de la media (superindividuos), evitando que dominen al resto y provocando así una convergencia prematura. Una de las técnicas más comunes consiste en la penalización de la función de evaluación, donde se penaliza la evaluación de ciertos individuos. Esta penalización puede ser estática o dinámica:

   a) En la penalización estática [Michalewicz y Schoenauer, 1996], se requieren unos coeficientes fijos establecidos por el diseñador del AG, por lo que la función de evaluación pasa a tener el siguiente aspecto:

   $$eval(\vec{x}) = f(\vec{x}) + W * penalizacion(\vec{x})$$

   donde f es la función objetivo, $penalizacion(\vec{x})$ es la función que penaliza al individuo y W es el coeficiente fijo.

   b) En la penalización dinámica, el parámetro W estático es reemplazado por un parámetro dinámico, $W(t)$, que es modificado en el tiempo. En el método propuesto por Joines y Houck [Joines y Houck, 1994], los individuos son evaluados (en la generación t) por la siguiente función:

   $$eval(\vec{x}) = f(\vec{x}) + (C * t)^{\alpha} * penalizacion(\vec{x})$$

donde $W(t) = (C * t)^{\alpha}$, con C y α constantes, y la penalización aumenta con la evolución del AG.

Existen otras técnicas diseñadas bajo este concepto, como las propuestas por Michalewicz y Attia [Michalewicz y Attia, 1994], con su método llamado *Genocop II* o por Bean y Hadj-Alouane [Bean y Hadj-Alouane, 1992], donde $W(t)$ es diseñada a partir de la información disponible sobre el espacio de búsqueda.

Otros métodos de adaptación de la función de evaluación son los propuestos por [Eiben y Ruttkay, 1996, Smith y Tate, 1993, Eiben y der Hauw, 1997].

3. Adaptación en el mecanismo de selección. Existen métodos de selección cuyos parámetros pueden ser facilmente adaptados, aunque en [Eiben et al., 1999] se indica que no se realiza con frecuencia la adaptación de este componente. Por ejemplo, en el método Linear Ranking Selection, donde los individuos de la población son ordenados en función de su bondad y se asigna una probabilidad de reproducción a cada uno dependiendo de su orden *rank(x_i)*, es posible variar la presión selectiva del AG modificando el valor de η_{min}, que especifica el número de copias esperadas del peor individuo en el intervalo [0,1].

4. Adaptación de los operadores genéticos. En este caso, diferentes operadores de cruce o de mutación son utilizados durante la ejecución del AG. En [Herrera et al., 1995b, 1996] se propone la utilización de diferentes operadores de cruce basados en el uso de conectivos difusos durante la ejecución del AG. Los autores proponen utilizar, en primer lugar, operadores que produzcan un alto nivel de diversidad en la población y, posteriormente, utilizar otros que produzcan el suficiente nivel de diversidad que permitan que se alcance la convergencia. Así, se presentan un conjunto de operadores de cruce dinámicos basados en el uso de t-normas, t-conormas y funciones promedio.

5. Adaptación de los parámetros de control. Los valores de los parámetros del AG son modificados durante la ejecución del mismo. Por tanto, pueden ser ajustados dinamicamente el tamaño de la población, N, la probabilidad de selección, P_s, la probabilidad de cruce, P_c, y la probabilidad de mutación, P_m. En la sección 3.3 se describe con mayor detalle este tipo de adaptación, ya que dentro de este aspecto se enmarca uno de los objetivos de esta tesis doctoral.

3.2.2. Tipos de adaptación

Respondiendo a la pregunta *¿cómo se realiza el cambio o el ajuste?* y, por tanto, atendiendo al *tipo de adaptación* utilizado en el algoritmo, los AGAs se pueden clasificar de la siguiente forma:

1. Control determinístico (deterministic parameter control), donde los valores de los parámetros son modificados atendiendo a alguna regla determinística, la cual los modifica sin utilizar ningún tipo de realimentación del estado de la búsqueda. Normalmente se utilizan esquemas en los que los parámetros varían en función del tiempo, es decir, se cambia un parámetro p por una función $p(t)$, donde t es el número de iteraciones. Sin embargo, si encontrar los parámetros óptimos estáticos para un problema dado es una tarea difícil, no más fácil es encontrar una función $p(t)$ para dicho problema. En [Bäck, 1991] se muestra una función lineal para controlar el decremento de la probabilidad de mutación.

2. Control adaptativo (adaptive parameter control). Con este método los valores de los parámetros son modificados atendiendo al estado de la búsqueda de la solución al problema, existiendo por tanto una realimentación que es utilizada para determinar la magnitud del cambio. Un ejemplo bien conocido de este tipo de adaptación es la 'regla del éxito 1/5' de Rechenberg's [Rechenberg, 1973]. También en Srinivas y Patnaik [1994] se muestra cómo los valores los parámetros se modifican en función de la evaluación de cada individuo y de la evaluación del resto de individuos de la población. Otro ejemplo que me gustaría resaltar es el control de los parámetros mediante controladores borrosos. Estos métodos reciben el nombre de Algoritmos Genéticos Adaptativos Borrosos (Fuzzy Adaptive Genetic Algorithms, FAGAs) y utilizan un controlador borroso (Fuzzy Logic Controller) para modificar los valores de los parámetros durante la ejecución del AG [Herrera y Lozano., 1996, Herrera y Lozano, 2003]. Como se muestra en la figura 3.1, el controlador tiene como entradas: ciertas medidas que indican el grado de satisfacción del comportamiento del AG, como por ejemplo, la convergencia y, los valores de los parámetros del AG. Como salida tiene los nuevos valores de los parámetros.

3. Control auto-adaptativo (self-adaptive parameter control). La idea de evolución sobre evolución puede ser utilizada para implementar la auto-adaptación de los parámetros. Ahora, los parámetros que se desean adaptar son codificados dentro de los individuos de la población y evolucionan junto a ellos,

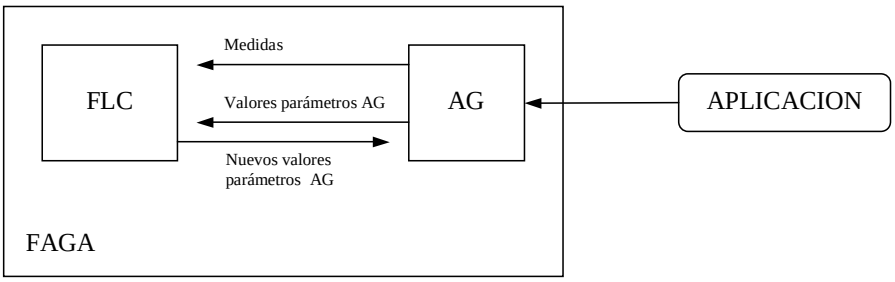

Figura 3.1: *Modelo de funcionamiento de un FAGA.*

aplicándoles también los operadores de cruce y mutación. Estos parámetros codificados no afectarán directamente a la evaluación de los individuos pero los "mejores" valores conllevarán a "mejores" individuos, los cuales tendrán mayores probabilidades de sobrevivir y producir descendencia. Por ejemplo, si se utiliza una codificación binaria, se puede añadir un gen extra p^i al final de la cadena de bits que representa a cada individuo de la población. El gen puede representar, por ejemplo, la probabilidad de mutación para todos los genes de la cadena y evolucionará también junto con la solución [Bäck y Schütz, 1996, Tuson y Ross, 1998].

3.2.3. Niveles de adaptación

Los cambios que se produzcan en el AG pueden afectar a un gen de un concreto individuo, a un individuo por completo, a toda la población de individuos o a la función de evaluación. Así, y respondiendo a la pregunta *¿a qué nivel se realiza el cambio o el ajuste?* y, por tanto, atendiendo al nivel en el cuál se realiza la adaptación, los AGAs se pueden clasificar de la siguiente forma:

1. Adaptación a nivel del entorno (Environment Level Adaptation), cuando se ajustan parámetros que afectan a la respuesta del entorno sobre un individuo. Esto sucede, por ejemplo, cuando se ajustan los coeficientes de penalización en la función de evaluación.

2. Adaptación a nivel de la población (Population Level Adaptation), cuando se ajustan parámetros globales cuyos valores afectan a todos los individuos de la población. La adaptación de esos parámetros globales se realiza normalmente de forma determinística o adaptativa.

Un ejemplo representativo de un método de control adaptativo a nivel de la población es la 'regla del éxito 1/5' de Rechenberg [Rechenberg, 1973]. Si la codificación de los individuos es real, se puede realizar una perturbación de sus valores mediante un valor aleatorio. Frecuentemente, se realiza mediante una distribución Gaussiana $N(0,\sigma)$, con media cero y desviación estándar σ. Por tanto, si se tiene una población donde cada individuo es representado como un vector de números reales $X = (x_1, \ldots, x_N)$ y se utiliza un operador de mutación guasiano, las mutaciones se realizan reemplazando los componentes de cada vector X de la población, pasando a ser $x_i = x_i + N(0,\sigma)$. Si la adaptación de σ se produce cada n generaciones se tiene:

Si (t **mod** n = 0) **entonces**

$$\sigma(t) = \begin{cases} \sigma(t-n)/c & \text{si } a > \frac{1}{5} \\ \sigma(t-n)c & \text{si } a < \frac{1}{5} \\ \sigma(t-n) & \text{si } a = \frac{1}{5} \end{cases}$$

si no

$$\sigma(t) = \sigma(t-1)$$

donde a es la frecuencia relativa de mutaciones exitosas medida sobre el número de generaciones y c un valor en el intervalo [0.817,1] [Bäck, 1996].

3. Adaptación a nivel del individuo (Individual Level Adaptation), cuando se ajustan parámetros en un individuo y cuyos valores afectan únicamente a ese individuo. El método de control adaptativo a nivel de la población de la 'regla del éxito 1/5' puede ser modificado para que actúe a nivel del individuo, dando lugar a un método de control auto-adaptativo. En este caso, cada individuo de la población tiene asociada una σ diferente, la cual es codificada dentro del mismo, por lo que cada individuo es representado como un vector de números reales de la forma $X = (x_1, \ldots, x_n, \sigma)$. Las mutaciones se realizan reemplazando los componentes de cada vector X de la población, pasando a ser $x_i' = x_i + N(0,\sigma')$, siendo $N(0,\sigma')$ una distribución normal, con media cero y desviación estándar $\sigma' = \sigma e^{N(0,\tau_0)}$ (τ_0 es un parámetro del método).

4. Adaptación a nivel del componente (Component Level Adaptation), cuando se ajustan parámetros locales que afectan a algún gen o genes de los individuos en la población. El método de control adaptativo a nivel de la población

de la 'regla del éxito 1/5' también puede ser modificado para que actúe a nivel del componente, dando lugar a un método de control auto-adaptativo a nivel del componente [Eiben et al., 1999]. En este caso, cada componente de cada individuo de la población tiene asociada una σ diferente y es representado como un vector de números reales $X = (x_1, \ldots, x_n, \sigma_1, \ldots, \sigma_n)$. Las mutaciones se realizan reemplazando cada componente de cada individuo, pasando a ser $x'_i = x_i + N(0, \sigma'_i)$, donde $N(0, \sigma'_i)$ es una distribución normal con media cero y desviación estándar $\sigma'_i = \sigma_i e^{N(0,\tau_0)}$ (τ_0 es es un parámetro del método).

Por lo general, el nivel al que se realiza el cambio depende del componente del AG donde se produce el mismo. Un cambio en la probabilidad de mutación puede afectar a un gen, a un individuo o la población entera, dependiendo de la implementación utilizada. Sin embargo, un cambio en los coeficientes de penalización de la función de evaluación afecta a la respuesta del entorno sobre cada individuo. Así, la clasificación atendiendo al nivel en que se realiza la adaptación puede considerarse secundaria frente a las anteriores, como se indica en [Eiben et al., 1999].

Por tanto, los principales criterios para clasificar los métodos de adaptación de los parámetros son atendiendo al parámetro que se cambia y a cómo se realiza el cambio, tratándose de una clasificación bi-dimensional que abarca los principales mecanismos adaptativos de los parámetros, como se muestra en la siguiente sección.

3.3. Adaptación de los parámetros de control

Diferentes valores de los parámetros pueden ser los óptimos en distintos estados del proceso evolutivo para obtener un buen equilibrio en la relación explotación/exploración. Por esta razón, algunos autores han sugerido la utilización de técnicas adaptativas, donde los valores de los parámetros del AG son modificados durante la ejecución del mismo. A continuación, se muestran los principales métodos de adaptación de los parámetros de control.

3.3.1. Adaptación de la probabilidad de mutación

Los principales mecanismos adaptativos para controlar la probabilidad de mutación durante la ejecución de un AG son los siguientes:

- **Control determinístico.** El valor del parámetro es modificado atendiendo a alguna regla determinística, la cual lo modifica sin utilizar ningún tipo de realimentación del estado de la búsqueda. Normalmente se utilizan esquemas donde el parámetro varía en función del tiempo, es decir, se cambia el parámetro p_m por una función $p_m(t)$, donde t es el contador de las iteraciones.

  Uno de los esquemas más utilizados consiste en decrementar el valor de la probabilidad de mutación durante la ejecución del AG [Fogarty, 1989, Bäck y Schütz, 1996]. Este esquema permite realizar una exploración en la etapa inicial de la ejecución y, posteriormente, una explotación de la información. Por otro lado, Hesser y Männer [Hesser y Männer, 1991] establecieron, para una represenación binaria de los individuos, la función:

$$p_m(t) = \sqrt{\frac{\alpha}{\beta}} \frac{exp(-\gamma \frac{t}{2})}{\lambda \sqrt{L}}$$

  donde α, β, γ son constantes, λ es el tamaño de la población y L la longitud de los individuos.

  La función determinada por Bäck y Schütz [Bäck y Schütz, 1996] fue:

$$p_m(t) = \ (2 + \tfrac{L-2}{T} t)^{-1} \quad \text{si } 0 \leq \text{t} \leq \text{T}$$

  que obliga a que $p_m(0)$=0.5 y $p_m(T) = 1/L$, siendo L la longitud de los individuos y T el número máximo de evaluaciones realizadas.

  Independientemente del tipo de codificación utilizado en el AG, en [Herrera y Lozano, 2003] los autores utilizan la siguiente función lineal, siguiendo la idea presentada en [Bäck y Schütz, 1996], para controlar el decremento de la probabilidad de mutación:

$$p_m(t) = \ p_m^h - \tfrac{p_m^h - p_m^l}{T} t \quad 0 \leq \text{t} \leq \text{T}$$

  donde $p_m(0) = p_m^h$, $p_m(T) = p_m^l$ y T es el número máximo de generaciones. Los autores utilizan los valores de $p_m(0)$=0.01 y p_m(T)=0.001

- **Control adaptativo.** En este caso, el valor del parámetro es modificado atendiendo al estado de la búsqueda de la solución al problema, existiendo por tanto una realimentación. Se pueden distinguir las siguientes técnicas:

  - **Control adaptativo a nivel del individuo.** En [Srinivas y Patnaik, 1994] se muestra un método donde la p_m se modifica en función de la evaluación de cada individuo y de la evaluación del resto de individuos de la población. Cada individuo de la población tiene asociada una probabilidad de mutación, la cual varía según la siguiente fórmula (para un problema de maximización):

$$p_m = \begin{cases} k_1 \frac{f_{max} - f'}{f_{max} - \overline{f}} & \text{si } f' \geq \overline{f}, \quad k_1 \leq 1 \\ k_2 & \text{si } f' < \overline{f}, \quad k_2 \leq 1 \end{cases}$$

    donde f' es la evaluación del individuo, f_{max} es la evaluación del mejor individuo en la población y $\overline{f}$ es la evaluación media de la población. Esta técnica protege a los individuos con una mayor adaptación, ya que a los mejores individuos se les aplican valores pequeños de p_m, mientras que a los peores se les aplican valores elevados. A su vez, se incrementa la p_m cuando la población tiende a un óptimo local y se decrementa cuando la población está dispersa en el espacio de soluciones. Los valores de k_1 y k_2 deben ser menor a 1.0 para que $p_m \in [0,1]$; en [Srinivas y Patnaik, 1994] los autores aplican los valores de k_1=0.5 y k_2=0.5.

  - **Control adaptativo a nivel de la población.** En [Herrera y Lozano, 2003] se muestra cómo utilizar un controlador borroso para mejorar el comportamiento de un AG. En concreto se describe un Algoritmo Genético Adaptativo Borroso (Fuzzy Adaptive Genetic Algorithm, FAGA) que integra un controlador borroso para adaptar la p_m durante la ejecución del AG. Este sistema es denominado *Fuzzy Adaptive Mutation Probability (GA-FAMP).*

    El controlador borroso se dispara cada G generaciones y la p_m permanece fija durante ese tiempo. El controlador, para hallar la nueva p_m, tiene en cuenta el valor de la p_m utilizado durante las últimas G generaciones y

la mejora alcanzada sobre el mejor individuo de la población. Una vez calculado el nuevo valor, se utiliza durante las próximas G generaciones.

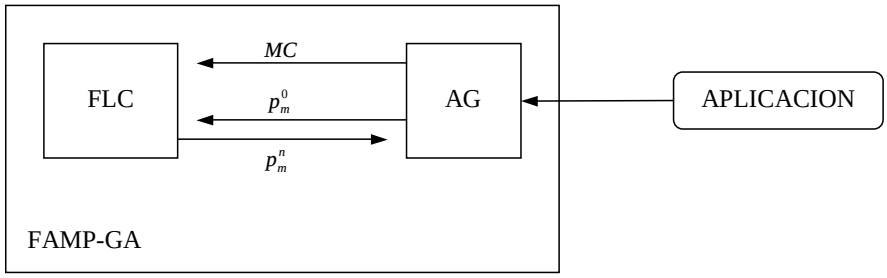

Figura 3.2: *Modelo de funcionamiento de un GA-FAMP.*

Como se muestra en la figura 3.2, el controlador borroso tiene dos entradas:

1. la probabiliad de mutación actual, p_m^0. Los autores consideran que este valor debe estar comprendido en el intervalo [0.001, 0.01], para una población de 60 individuos. El conjunto de etiquetas asociadas con p_m^0 es {Bajo, Medio, Alto}, como se muestra en la figura 3.3.
2. la medida de la convergencia, $MC = f_b^c/f_b^0$ (para un problema de minimización), donde f_b^c es la evaluación del mejor individuo encontrado hasta el momento y f_b^0 es la evalución del mejor individuo encontrado antes de las últimas G generaciones. Si se adopta una estrategia elitista, el valor de MC estará comprendido en el intervalo [0,1]. Si la MC es alta entonces la convergencia es alta, es decir, no existe mejora durante las últimas G generaciones. Por el contrario, si la MC es baja, el AG ha encontrado un nuevo individuo que mejora al anterior. El conjunto de etiquetas asociadas con la MC es {Bajo, Alto}, como se muestra en la figura 3.3.

Como salida, el controlador borroso tiene el nuevo valor de p_m, p_m^n, que es utilizado durante las próximas G generaciones. El conjunto de etiquetas asocidas con p_m^n es {Bajo, Medio, Alto}, como también se muestra en la figura 3.3.

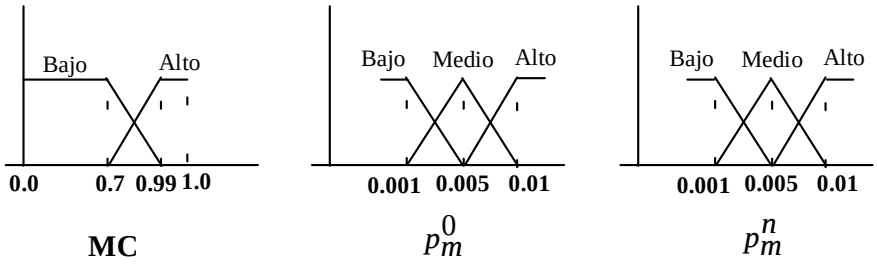

Figura 3.3: *Significado de los conjuntos asociados en GA-FAMP.*

Tabla 3.1: *Base de reglas para el control de* p_m.

Regla	**MC**	(p_m^0)	(p_m^n)
1	Alto	Bajo	Medio
2	Alto	Medio	Alto
3	Alto	Alto	Bajo
4	Bajo	Bajo	Bajo
5	Bajo	Medio	Bajo
6	Bajo	Alto	Medio

Las reglas borrosas del controlador que relacionan las entradas y la salida son las mostradas en la tabla 3.1. Como puede apreciarse, a través de las reglas 1, 2 y 4-6 se indica que la p_m se debe decrementar cuando exista progreso en la evolución y se debe incrementar cuando no exista mejora. Una p_m elevada puede ser la causa de un bajo rendimiento en el AG, ya que evita la convergencia para obtener mejores individuos. Para tratar de evitar esta circunstancia se introduce la regla número 3, que propone una p_m baja, cuando la MC es alta y la p_m^0 es alta.

- **Control auto-adaptativo.** Con esta técnica el valor del parámetro p_m es codificado dentro de los individuos de la población y evoluciona junto a ellos.

  - **Control auto-adaptativo a nivel del individuo.** En el caso de utilizar una representación binaria de los individuos, se añade un gen extra p_m^i al final de la cadena de bits, C_i, que representa a cada uno de los individuos de la población. p_m^i indica la probabilidad de mutación para todos los genes de cada C_i y evolucionará junto con la solución [Bäck y

Schütz, 1996, Tuson y Ross, 1998]. Los valores de p_m^i pueden variar en el intervalo $[p_m^l, p_m^h]$. Para mutar los genes en un individuo C_i se procede de la siguiente forma:

1. Se aplica una meta-mutación sobre p_m^i, obteniendo ρ_m^i. Esto se lleva a cabo eligiendo un valor aleatorio en el intervalo $[p_m^i - \delta, p_m^i + \delta]$, donde δ es un parámetro de control que ha de ser establecido por el diseñador del AG.
2. Se mutan los genes de C_i utilizando la probabilidad de mutación obtenida anteriormente, ρ_m^i.
3. Los genes mutados, incluyendo el valor ρ_m^i, se incluyen de nuevo en el individuo.

El operador de cruce se aplica únicamente sobre la cadena de bits que representa a cada individuo y no tiene efecto sobre p_m^i. La descendencia resultante recibe el valor p_m^i de uno de sus padres. Los valores iniciales de p_m^i son generados aleatoriamente dentro del intervalo $[p_m^l, p_m^h]$.

3.3.2. Adaptación del tamaño de la población

El tamaño idóneo de la población en un AG ha sido objeto de numerosas investigaciones ya que puede ser crítico en muchas aplicaciones. Si el tamaño de la población es pequeño se corre el riesgo de no cubrir adecuadamente el espacio de búsqueda y que la convergencia sea muy rápida, mientras que si es demasiado elevado puede conllevar problemas relacionados con el excesivo costo computacional.

Uno de los primeros en adaptar el tamaño de la población fue Baker [Baker, 1987a] que observó cómo se producía una rápida convergencia cuando un individuo o un pequeño grupo de individuos daba un gran número de descendientes. Dado que la población de individuos es finita, un gran número de descendientes de un mismo individuo implica una menor descendencia para el resto de la población. Cuando existen muchos individuos que no tienen ningún descendiente se produce una convergencia prematura y una pérdida de la diversidad. Para detectar una convergencia prematura, antes de que todo el material genético sea perdido, Baker propuso una medida, denominada *percent involvement*, que indica el porcentaje de la población actual que contribuye a crear la descendencia en la próxima generación. El método propuesto implica una población dinámica y garantiza una cota inferior sobre el

percent involvement. Esto se lleva a cabo añadiendo individuos a la población hasta que se obtiene un valor aceptable para la medida *percent involvement*. Por otro lado, en periodos de convergencia lenta se disminuye el tamaño de la población.

Otro enfoque es presentado en [Arabas et al., 1994], donde se introduce el concepto de *edad* de un individuo, que equivale al número de generaciones que dicho individuo permanece vivo. El algoritmo propuesto, denominado GAVaPS (Genetic Algorithm with Varying Population Size), no utiliza ningún mecanismo de selección ya que es sustituido por el concepto de *edad*. En el momento que existe un nuevo individuo, se le asigna un tiempo de vida, *lifetime*. Este individuo deja de pertenecer a la población cuando su edad alcanza el valor de su *lifetime*. Para calcular el *lifetime* de cada individuo se tienen en cuenta los siguientes factores: la evaluación máxima y mínima de los individuos en la población actual, la evaluación media y la evaluación máxima y mínima hallada hasta el momento. Los individuos cuya evaluación esté por encima de la media de la población deben tener un valor elevado de su *lifetime*. Diferentes propuestas para el cálculo del *lifetime* han sido implementadas y comparadas por los autores. Sin embargo, ninguna de ellas ha destacado como la mejor con respecto a los resultados y a su coste computacional. Así, en el artículo se indica que el cálculo del *lifetime* para cada individuo es un problema abierto para futuras investigaciones.

Smith [Smith, 1993] propuso un algoritmo que adapta el tamaño de la población con respecto a la probabilidad de error de selección. También en [Hinterding et al., 1996] los autores experimentaron con un AGA que estaba constituido por tres subpoblaciones y, a intervalos regulares, los tamaños de las poblaciones eran adaptados teniendo en cuenta el estado de la búsqueda. En [Lobo y Lima, 2005, 2007] se muestra una revisión de los distintos esquemas utilizados en AGs para adaptar el tamaño de la población.

3.3.3. Adaptación de la probabilidad de cruce

En [Wilson, 1986], donde los autores utilizan AGs para sistemas clasificadores, se indica que la variación de la probabilidad de cruce durante la ejecución del AG puede mejorar el comportamiento del mismo. El enfoque está basado en la utilización de una medida de entropía sobre la población de individuos. La P_c se incrementa ligeramente si la entropía disminuye, y se decrementa si la entropía aumenta o se mantiene.

Por otro lado, en [Booker, 1987] se indica que el operador de cruce es la clave para resolver el problema de la convergencia prematura. Se propone variar la P_c basándose en la utilización de la medida *percent involvement*, definida en [Baker, 1987a] y ya comentada en el apartado 3.3.2. En cada cambio del porcentaje de la medida *percent involvement* le sucede un cambio igual (en porcentaje) pero opuesto en la P_c.

En [Srinivas y Patnaik, 1994] se muestra un método de control adaptativo, donde la P_c se modifica en función de la evaluación de los individuos que actúan como padres y de la evaluación del resto de individuos de la población. La probabilidad de cruce varía según la siguiente fórmula (para un problema de maximización):

$$p_c = \begin{cases} k_3 \frac{f_{max} - f}{f_{max} - \overline{f}} & \text{si } f \geq \overline{f}, \quad k_3 \leq 1 \\ k_4 & \text{si } f < \overline{f}, \quad k_4 \leq 1 \end{cases}$$

donde f es la mayor de las evaluaciones de los individuos a ser cruzados, f_{max} es la evaluación del mejor individuo en la población y $\overline{f}$ es la evaluación media de la población.

Los valores k_3 y k_4 deben ser menores o iguales a 1.0 para que $p_c \in [0,1]$. Los autores aplican los valores k_3=1.0 y k_2=1.0, lo cual asegura que el cruce se lleva a cabo cuando los individuos seleccionados para ser cruzados tienen una evaluación menor o igual a la evaluación media. La P_c disminuye a medida que la evaluación de uno de los padres a ser cruzados tiende a f_{max}, llegando a ser 0.0 en tal caso.

3.3.4. Adaptación de las probabilidades de cruce y mutación

También es posible realizar una adaptación simultáneamente de varios parámetros de un AG, aunque la mayoría de los estudios sobre adaptación consideran solamente un único parámetro. Esto es debido, probablemente, a que es más fácil extraer conclusiones sobre los resultados al no existir interacción entre varios parámetros [Eiben et al., 1999]. A continuación, se indican los principales estudios que se han realizado para adaptar simultáneamente las probabilidades de cruce y mutación.

En [Srinivas y Patnaik, 1994], se recomienda la adaptación de P_c y P_m para mantener la diversidad en la población de individuos y sostener la capacidad de convergencia en el AG. Así, se propone un AGA donde P_c y P_m son adaptadas en función de las evaluaciones de los individuos. Cada individuo de la población tiene

su propia P_c y P_m, que atienden a las siguientes fórmulas:

$$P_m = \begin{cases} k_1 \frac{f_{max} - f'}{f_{max} - \overline{f}} & \text{si } f' \geq \overline{f}, \quad k_1 \leq 1 \\ k_2 & \text{si } f' < \overline{f}, \quad k_2 \leq 1 \end{cases}$$

$$P_c = \begin{cases} k_3 \frac{f_{max} - f}{f_{max} - \overline{f}} & \text{si } f \geq \overline{f}, \quad k_3 \leq 1 \\ k_4 & \text{si } f < \overline{f}, \quad k_4 \leq 1 \end{cases}$$

donde f' es la evaluación del individuo, f es la mayor de las evaluaciones de los individuos a ser cruzados, f_{max} es la evaluación del mejor individuo en la población y $\overline{f}$ es la evaluación media de la población. Esta técnica protege a los individuos con una mayor adaptación. Los valores de P_c y P_m se incrementan cuando la población tiende a un óptimo local y se decrementan cuando la población está dispersa en el espacio de soluciones. Los valores de k_1, k_2, k_3, k_4 deben ser menores o iguales a 1.0 para que P_c, $P_m \in$ [0,1]. Los autores utilizan: k_1=0.5, k_2=0.5, k_3=1.0, k_4=1.0.

En [Li et al., 1992] se establecen dos funciones de diversidad que se utilizan para adaptar los parámetros P_c y P_m. Una de las funciones mide la diversidad entre los individuos de la población, mientras que la otra mide la diversidad entre los genes de los individuos. El procedimiento de implementación del AGA se divide en tres etapas: inicialización, búsqueda y refinamiento.

En [Davis, 1989] se muestra la implementación de un AG con codificación real, donde se utilizan dos tipos de operadores de cruce y tres tipos de operadores de mutación, que ofrecen una amplia gama de niveles de exploración y explotación. Cada operador tiene una asociada una probabilidad inicial de aplicación. Para cada reproducción, se selecciona un único operador de forma probabilística, teniendo en cuenta las probabilidades de los operadores. Mediante un proceso adaptativo se modifican las probabilidades de aplicación de los operadores, en función de la evaluación de los individuos creados por dichos operadores. A aquellos operadores que generan una buena descendencia se les asignan probabilidades de aplicación elevadas, por lo que serán utilizados más frecuentemente. Por el contrario, los operadores que generan descendencia cuya evaluación sea inferior a la de los padres se utilizan con menos frecuencia.

En [Julstrom, 1995] se presenta un modelo similar al anterior para AG estacionarios donde se adaptan las probabilidades de aplicación de los operadores después de la generación de cada descendencia.

3.3.5. Adaptación del tamaño de la población y de las probabilidades de cruce y mutación

Como se ha comentado en el apartado anterior, es posible realizar una adaptación simultánea de varios parámetros de un AG durante su ejecución. En [Lis y Lis, 1996] los autores adaptan el tamaño de la población y las probabilidades de mutación y cruce, utilizando para ello un AG paralelo.

3.4. Conclusiones

En este capítulo se han revisado los conceptos más importantes sobre Algoritmos Genéticos Adaptativos. Se ha mostrado una clasificación de los mismos basándose en los componentes que pueden ser adaptados, en los niveles a los que se puede producir la adaptación y en los tipos en los que se realiza dicha adaptación. Además, y dado que en esta tesis doctoral se propone un nuevo sistema de adaptación de parámetros, se ha considerado conveniente revisar las contribuciones más importantes recogidas en la literatura sobre la adaptación de dichos parámetros de control en un AG.

Una opción para encontrar la "mejor" configuración del AG para un problema dado es introducir algún tipo de adaptación dentro del algoritmo, con el fin de realizar dos búsquedas simultáneamente: la búsqueda de la solución al problema dado y la búsqueda de la "mejor" configuración del AG. Sin embargo, no está muy claro qué y cuantos parámetros considerar en la búsqueda, surgiendo cuestiones como:

> *¿es factible considerar todo el espacio de búsqueda del algoritmo y permitir a éste que realice la selección y los cambios en la representación de individuos y de los operadores?*
>
> *¿debería el algoritmo simultáneamente controlar las probabilidades de los operadores genéticos, el tamaño de la población y el método de selección?*
>
> *¿qúe coste computacional es aceptable?*

Por tanto, como puede apreciarse, este es un campo en el que todavía quedan aspectos por resolver e investigar, como el desarrollo de modelos para comparar algoritmos con y sin mecanismos adaptativos o la combinación de distintos tipos y niveles de adaptación en los parámetros, lo que conllevaría mejoras significativas en las soluciones encontradas a los problemas.

Capítulo 4

Sistemas Borrosos y Borroso-Genéticos

4.1. Introducción

El objetivo de este capítulo es describir brevemente los sistemas borrosos y borroso-genéticos, presentando una revisión de su estado del arte, ya que es uno de los escenarios donde se aplicará el sistema de optimización propuesto en esta tesis doctoral. La lógica borrosa fue introducida por L.A. Zadeh [Zadeh, 1965] en el año 1965, la cual puede ser concebida como una extensión de los sistemas lógicos clásicos, proporcionando un marco ideal para trabajar con el problema de la representación del conocimiento en entornos de incertidumbre e imprecisión ya que define un tipo de conjuntos en los que se permite una transición gradual entre la completa pertenencia y la completa exclusión.

Los sistemas borrosos basados en reglas han sido utilizados para modelar problemas humanos donde el modo de representar el conocimiento humano es con el uso de reglas IF-THEN. Una de las aplicaciones más destacadas de los sistemas borrosos basados en reglas son los sistemas de control borroso, en los que las bases de reglas proporcionan una solución al control de un determinado sistema.

Dado que no existe ningún módulo en los sistemas borrosos basados en reglas que se encargue de la adquisición de conocimiento, se han realizado numerosos trabajos en donde se indica como integrarlos con un proceso de aprendizaje evolutivo. Estos sistemas reciben el nombre de sistemas borroso-genéticos [Magdalena y Velasco,

1996, Herrera y Magdalena, 1997, Cordón et al., 2001, Herrera, 2004, Cordón et al., 2004, Herrera, 2005]. Por tanto, un sistema borroso-genético es un sistema borroso al que se le añade un proceso de aprendizaje evolutivo para automatizar su diseño. Los principales enfoques que existen en el aprendizaje en dichos sistemas se conocen como: Pittsburgh, Michigan e IRL.

En los siguientes apartados presentan brevemente los conceptos básicos sobre sistemas borrosos basados en reglas, controladores borrosos, sistemas borroso-genéticos y por último, sistemas borroso-genéticos basados en el enfoque de Pittsburgh, ya que es uno de los escenarios en los que se aplica el sistema de optimización de parámetros propuesto en esta tesis.

4.2. Sistemas borrosos basados en reglas

Los sistemas borrosos basados en reglas han sido utilizados para modelar problemas humanos donde el modo de representar el conocimiento humano es con el uso de reglas IF-THEN. El cumplimiento del antecedente da lugar a la ejecución del consecuente y, por tanto, a la ejecución de una determinada acción. Así, los sistemas borrosos basados en reglas utilizan lógica borrosa en sus variables antecedentes y consecuentes y un conjunto de reglas lingüísticas con expresiones del tipo IF-THEN. La figura 4.1 muestra la estructura básica de un sistema borroso basado en reglas, consistente en una etapa de entrada, un motor de inferencias, una base de conocimiento y una etapa de salida.

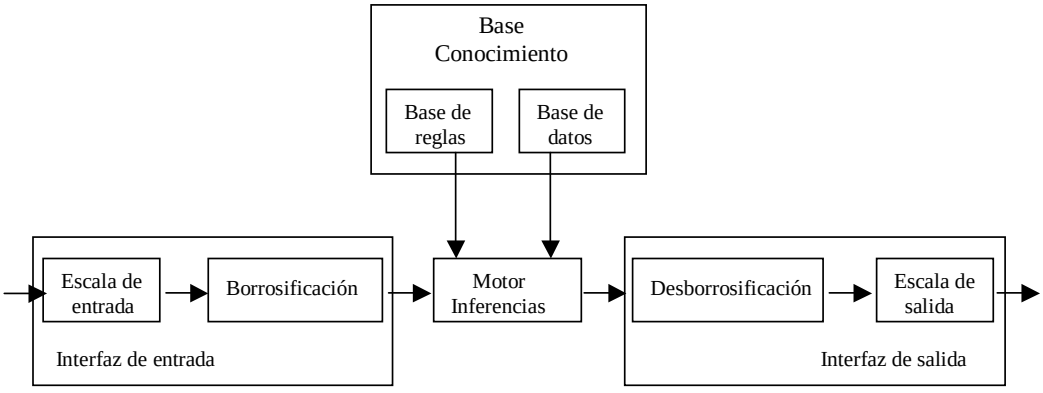

Figura 4.1: *Estructura de un sistema borroso basado en reglas.*

- Interfaz de entrada. Está compuesto por una etapa de escalado y una etapa de borrosificación. El escalado a la entrada adapta el rango de los valores reales de las variables al universo del discurso de las variables lingüísticas. La borrosificación transforma los valores de las variables de entrada en información borrosa permitiendo que el sistema pueda trabajar con datos reales.

- Motor de inferencias. Es el que infiere los valores de las variables borrosas de salida a partir de la base de conocimiento y del estado del sistema, definido como un conjunto de valores de las variables borrosas.

- Base de Conocimiento. Es la que incorpora el conocimiento del sistema. Cada base de conocimiento tiene dos componentes, una base de datos y una base de reglas. La base de datos contiene información sobre las variables borrosas de entrada y salida, sobre las funciones de pertenencia y sobre las funciones de escalado. La base de reglas contiene el conjunto de reglas difusas de actuación.

- Interfaz de salida. Está compuesto por una etapa de desborrosificación y una etapa de escalado a la salida. La desborrosificación adapta la salida del sistema borroso a valores numéricos normalizados de las variables de salida. La función de escalado transforma los valores normalizados a los rangos de las variables.

Existen dos modelos de sistemas borrosos basados en reglas para problemas de ingeniería:

- Modelo de Mamdani [Mamdani y Assilian, 1975]. En este modelo se utilizan variables lingüísticas, tanto en el antecedente como en el consecuente de las reglas de conocimiento, siendo su expresión de la siguiente forma:

  $$\text{Si } (X_1 \text{ es } A_1) \text{ y } (X_2 \text{ es } A_2) \text{ y } \ldots\ldots (X_n \text{ es } A_n) \text{ ENTONCES (Y ES B)}$$

  donde X_i son las variables de entrada, A_i son conjuntos borrosos definidos en el dominio de las variables de entrada, Y es la variable de salida y B es un conjunto borroso para la variable de salida.

- Modelo de Takagi, Sugeno y Kang (TSK) [Takagi y Sugeno, 1985]. En 1985 Takagi, Sugeno y Kang propusieron un nuevo modelo donde el antecedente de las reglas está compuesto por variables lingüisticas y el consecuente está representado por una función de las variables de entrada. La forma más habitual es aquella donde la función constituye una relación lineal de las variables del antecedente, como se muestra a continuación:

$$\text{Si } (X_1 \text{ es } A_1) \text{ y } (X_2 \text{ es } A_2) \text{ y } \ldots\ldots (X_n \text{ es } A_n) \text{ ENTONCES}$$
$$(Y = p_1 * X_1 + p_2 * X_2 + \ldots + p_n * X_n + p_0\ )$$

donde X_i son las variables de entrada, A_i son conjuntos borrosos de las variables de entrada, Y es la variable de salida y p_i son los parámetros de la relación lineal.

Las principales aplicaciones de los sistemas borrosos basados en reglas son el modelado borroso de sistemas, el control borroso y los clasificadores borrosos [Cordón et al., 2001]. A continuación se describen los sistemas de control borroso.

4.3. Controladores borrosos

Una de las aplicaciones más destacadas de los sistemas borrosos basados en reglas son los sistemas de control borroso (Fuzzy Logic Controller, FLC), en los que las bases de reglas proporcionan una solución al control de un determinado sistema.

Los controladores lógicos borrosos [Pedryck, 1989] pueden ser considerados como sistemas basados en conocimiento, que incorporan conocimiento humano a sus bases de conocimiento a través de un conjunto reglas de actuación y funciones de pertenencia [Magdalena y Velasco, 1996].

La estructura de un controlador lógico borroso se muestra en la figura 4.2 y está formado por los siguientes elementos:

- Interfaz de entrada. Compuesto por una etapa de escalado y una etapa de borrosificación. El escalado a la entrada adapta el rango de los valores reales de las variables al universo del discurso de las variables lingüísticas. La borrosificación transforma los valores de las variables de entrada en información borrosa, permitiendo que el sistema pueda trabajar con datos reales.

- Motor de inferencias. Infiere los valores de las variables borrosas de salida, a partir de la base de conocimiento y del estado del sistema, definido como un conjunto de valores de las variables borrosas.

- Base de Conocimiento. Es la que incorpora el conocimiento del sistema. Cada base de conocimiento tiene dos componentes, una base de datos y una base de

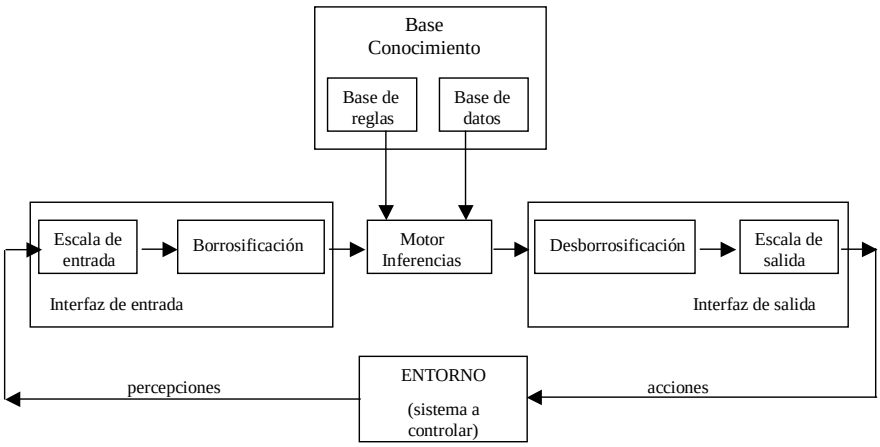

Figura 4.2: *Estructura de un controlador lógico borroso.*

reglas. La base de datos contiene información sobre las variables borrosas de entrada y salida, sobre las funciones de pertenencia y sobre las funciones de escalado. La base de reglas contiene el conjunto de reglas difusas de actuación.

- Interfaz de salida. Compuesto por una etapa de desborrosificación y una etapa de escalado a la salida. La desborrosificación adapta la salida del sistema borroso a valores numéricos normalizados de las variables de salida. La función de escalado transforma los valores normalizados a los rangos de las variables.

- Entorno o sistema a controlar. Es el entorno de actuación del controlador borroso. Las percepciones del estado del sistema se obtienen por medio de unos sensores y a través de unos actuadores se realizan las acciones inferidas por el controlador.

Como puede apreciarse, no existe ningún módulo en la estructura del controlador borroso que se encargue de la adquisición de conocimiento. Por tanto, la incorporación de éste se tiene que realizar a través de expertos. Sin embargo, es posible añadir un sistema que se encargue del aprendizaje. En el caso de utilizar un sistema de aprendizaje basado en AGs, estos sistemas reciben el nombre de sistemas borroso-genéticos [Magdalena y Velasco, 1996, Herrera y Magdalena, 1997, Cordón et al., 2001, Herrera, 2004], que añaden la evolución del conocimiento a los sistemas borrosos y, consecuentemente, la adquisición de nuevo conocimiento. A continuación se describen dichos sistemas.

4.4. Sistemas borroso-genéticos

Como se ha definido anteriormente, un sistema borroso-genético es un sistema borroso al que se le añade un proceso de aprendizaje evolutivo, ya que una de las principales desventajas de los sistemas borrosos basados en reglas es que no tienen capacidad de aprendizaje y, por tanto, es preciso emplear un proceso de aprendizaje para automatizar su diseño.

La definición automática de un sistema borroso basado en reglas puede ser visto como un problema de optimización o búsqueda y, como es bien sabido, los AGs proporcionan un mecanismo de búsqueda y optimización robusto basados en la evolución natural y la genética. Además de la capacidad búsqueda en espacios complejos, los AGs tienen una estructura de código genérica y unas caracteristicas de funcionamiento idóneas para incorcopar conocimiento a priori. En el caso de los sistemas borrosos basados en reglas este conocimiento a priori puede ser en la forma de variables lingüísticas, reglas borrosas, número de reglas, parámetros de las funciones de pertenencia, etc. Debido a estas capacidades se ha extendido el uso de AGs en el diseño de sistemas borrosos basados en reglas.

Es importante resaltar que el proceso de aprendizaje genético tiene el objetivo de diseñar u optimizar la base de conocimiento. Por tanto, un sistema borroso-genético es un método de diseño de sistemas borrosos basados en reglas, que incorpora un sistema evolutivo para lograr la generación automática ó la modificación de una base de conocimiento (o de una parte de ella).

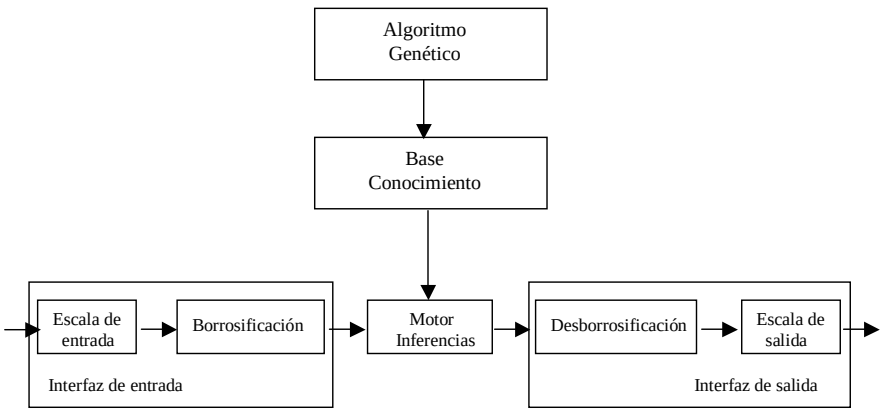

Figura 4.3: *Estructura de un sistema borroso-genético.*

A la hora de diseñar un sistema borroso-genético lo primero que se ha de decidir es qué parte de la base de conocimiento se desea que el algoritmo evolutivo optimice, ya que puede optimizarse la base de reglas, la base de datos o ambas. El espacio de búsqueda del algoritmo evolutivo es diferente dependiendo de la parte de la base de conocimiento que se quiera optimizar. En este punto es importante distinguir entre un proceso de aprendizaje y un proceso de ajuste o "tuning".

El proceso de ajuste o "tuning" versa sobre la optimización de un sistema borroso basado en reglas existente y asume una base de reglas predefinida. Tiene el objetivo de encontrar un conjunto de parámetros óptimos para las funciones de pertenencia y/o las funciones de escalado, es decir, los parámetros de la base de datos.

El proceso de aprendizaje constituye un método de diseño automático de algunos componentes de la base de conocimiento o de la base de conocimiento al completo (se parte de cero). Este proceso realiza una búsqueda más elaborada en el espacio de posibles bases de reglas o bases de conocimiento y no depende de un conjunto de reglas predefinido.

Como se indica en [Herrera, 2004], los modelos de ajuste siguen un patrón común, mostrándose las diferencias en los diferentes parámetros codificados, en los tipos de función de pertenencia, o en el tipo de sistemas difusos (Mamdani, TSK, ...), entre otros aspectos. Dentro de los modelos de aprendizaje de sistemas borrosos basados en reglas se pueden distinguir tres modelos básicos [Herrera, 2004, 2005]:

- el aprendizaje de una base de reglas (figura 4.4).

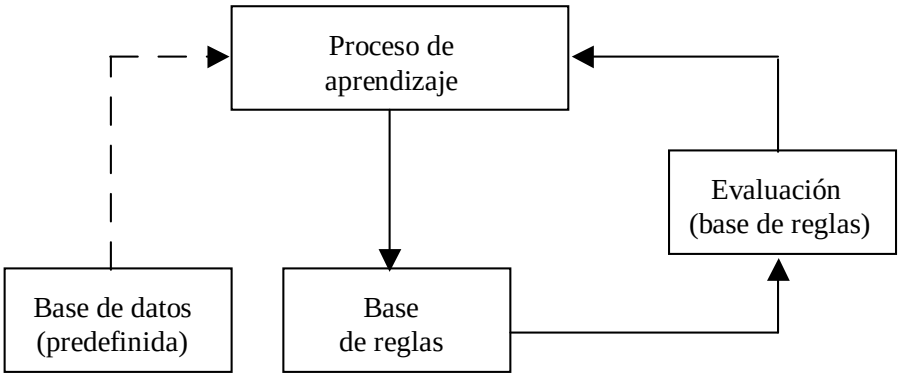

Figura 4.4: *Aprendizaje evolutivo de una base de reglas.*

- el aprendizaje de una base de conocimiento, donde conjuntamente se aprenden las funciones de pertenencia y la base de reglas (figura 4.5).

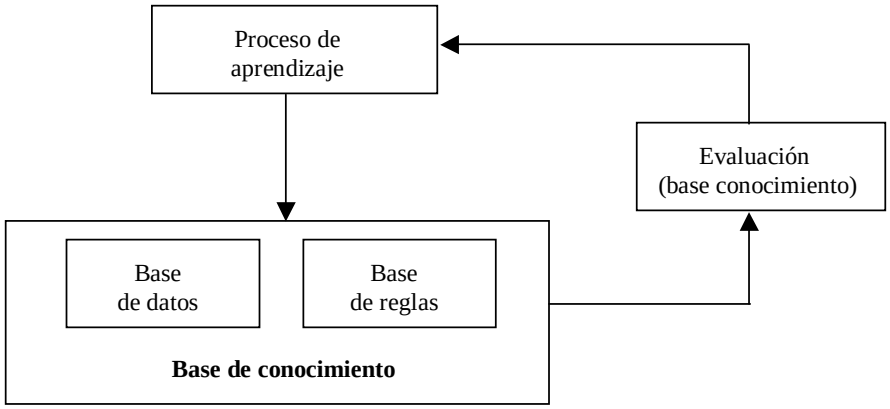

Figura 4.5: *Aprendizaje evolutivo de una base de conocimiento.*

- el aprendizaje de la base de datos mediante un AG y el uso de un algoritmo base para el aprendizaje de la base de reglas (figura 4.6).

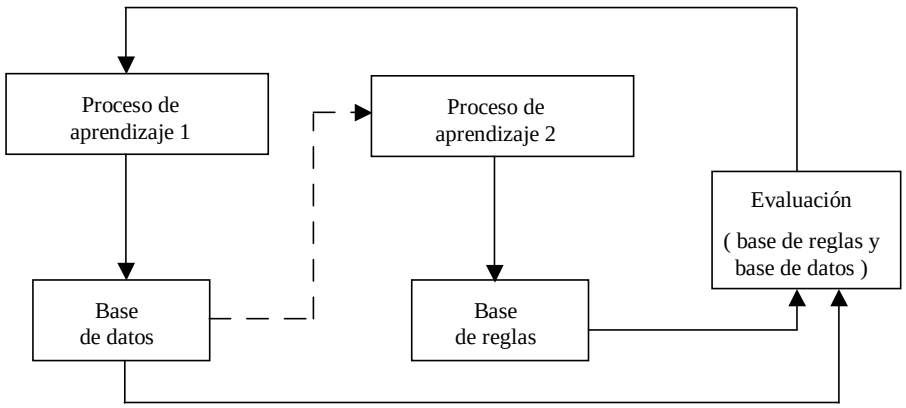

Figura 4.6: *Aprendizaje evolutivo de una base de datos.*

Por tanto, en el diseño de un sistema borroso-genético se ha de decidir si se realiza:

a) un ajuste genético de la base de datos, ajustando las funciones de escalado y/o de pertenencia.

- ajuste de las funciones de escalado. Las funciones de escalado que se aplican en las etapas de entrada y salida utilizan diversos parámetros (escalas lineales, límites, factor de contracción-dilatación) con el objeto de adaptar el rango de la variable con los valores normalizados. En este sentido, el ajuste genético permitiría realizar un ajuste sobre los parámetros de dichas funciones de escalado.
- ajuste de las funciones de pertenencia. En cuanto a las funciones de pertenencia asociadas a las variables borrosas, el algoritmo evolutivo puede actuar mediante el ajuste de parámetros que definen la forma de los conjuntos borrosos, mediante el cruce de conjuntos borrosos con otras variables o mediante la mutación de puntos que definen los conjuntos borrosos.

b) un aprendizaje genético de la base de reglas. En este caso, se asume un conjunto de funciones de pertenencia predefinido en la base de datos a las que las reglas se refieren. El algoritmo evolutivo evoluciona la base de reglas, bien trabajando con individuos que describen a un única regla (enfoque de Michigan) o a una base entera de reglas (enfoque de Pittsburgh).

c) un aprendizaje genético de la base de conocimiento. En este caso, el aprendizaje trata con un espacio de búsqueda heterogéneo que abarca diferentes representaciones genéticas como cromosomas de longitud variable, representación multi-cromosoma (donde los cromosomas se dividen en dos sub-cromosomas, uno codifica la base de datos y el otro la base de reglas) o cromosomas codificando reglas individuales en lugar de una base de conocimiento entera. El coste computacional crece a medida que crece la complejidad del espacio de búsqueda.

Dado que los procesos de aprendizaje genéticos pueden operar en dos clases diferentes de espacios: a) un espacio de reglas y b) un espacio de bases de reglas o bases de conocimiento enteras, existen distintos enfoques para integrar los sistemas de control borroso y los sistemas de aprendizaje basados en AGs. Principalmente existen 3 modelos para el aprendizaje en dichos sistemas [Magdalena y Velasco, 1996, Herrera y Magdalena, 1997, Cordón et al., 2001]:

- Enfoque de Pittsburgh, en el que los individuos de la población son bases de conocimiento completas o bases de reglas. El sistema evolutivo obtiene nuevas bases de conocimiento, las cuales son evaluadas actuando sobre el entorno y analizando su evolución.

- Enfoque de Michigan, donde los individuos de la población son reglas individuales de actuación de la base de conocimiento por lo que dicha base representa a la población entera. Este enfoque, utilizado frecuentemente como aprendizaje on-line, está considerado como un sistema clasificador que utiliza el reforzamiento mediante asignación de créditos y el descubrimiento de reglas para realizar el aprendizaje.

- Enfoque Iterative Rule Learning (IRL), fue propuesto en 1993 combinando los enfoques de Michigan y Pittsburgh [Venturini, 1993] y adaptado, posteriormente, a los sistemas borroso-genéticos basados en reglas [Cordón et al., 1999]. En este enfoque, como en el de Michigan, cada individuo de la población representa a una única regla pero contrariamente a éste, sólo se introduce en el conjunto final de reglas la mejor regla obtenida mediante AGs. Si el conjunto final de reglas generado cumple con los objetivos dados, el método de aprendizaje se detiene. En caso contrario, se realiza una nueva iteración. En IRL, los AGs obtienen una solución parcial en el aprendizaje (contrario a los enfoques anteriores) siendo necesario repetirlo un número de iteraciones hasta obtener un conjunto final de reglas.

Seguidamente, se presenta con más detalle cómo se realiza el aprendizaje evolutivo de una base de conocimiento sobre un controlador borroso basado en el enfoque de Pittsburgh dado que, de los tres modelos, es donde se centra el trabajo realizado en esta tesis.

4.5. Sistemas borroso-genéticos basados en el enfoque de Pittsburgh

Los sistemas borroso-genéticos basados en el enfoque de Pittsburgh utilizan bases de conocimiento completas o bases de reglas como individuos de la población. En este enfoque el sistema evolutivo obtiene nuevas bases de conocimiento o de reglas, que son evaluadas actuando directamente sobre el entorno y analizando su evolución.

En la figura 4.7 se muestra la estructura general de un sistema borroso-genético basado en el enfoque de Pittsburgh [Herrera y Magdalena, 1997]. Puede observarse cómo se realiza la integración del sistema de control borroso con el sistema a controlar y el sistema evolutivo.

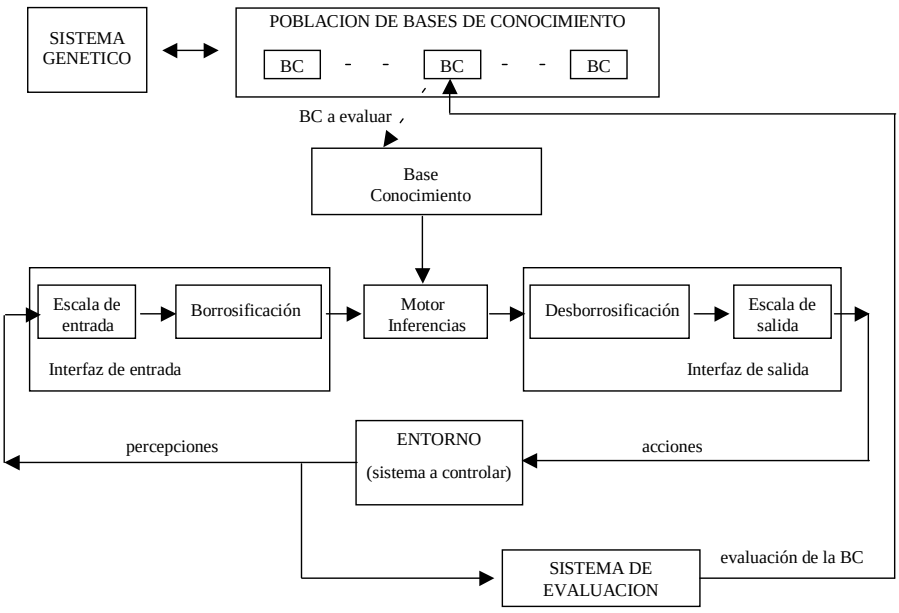

Figura 4.7: *Aprendizaje evolutivo de una base de conocimiento sobre un FLC basado en el enfoque Pittsburgh.*

En la estructura del sistema borroso-genético se pueden distinguir los siguientes elementos:

- Sistema borroso, formado por los interfaces de entrada y salida, el motor de inferencias y una base de conocimiento.
- Entorno o sistema a controlar, sobre el que actúa el sistema borroso a través de las percepciones y actuaciones (variables de contexto y de operación).
- Población de bases de conocimiento, que representan un conjunto de soluciones del sistema de control.
- Sistema de evaluación, que realiza una medida de la bondad o grado de adaptación de las bases de conocimiento al entorno.
- Sistema genético, que permite obtener nuevas bases de conocimiento o bases de reglas a partir de la población inicial. Los sistemas borroso-genéticos basados en este enfoque centran el proceso de aprendizaje sobre la base de reglas, aunque también es posible incorporar una parte de la base de datos en dicho proceso (funciones de pertenencia o escalado). En el caso que el proceso de

aprendizaje actúe sobre los dos componentes de una base de conocimiento, los individuos han de contener información sobre la base de reglas y sobre las funciones de pertenencia, lo cual requiere la utilización de una codificación multi-cromosoma, donde los cromosomas se dividen en dos sub-cromosomas, uno codifica la base de datos y el otro la base de reglas. A su vez, el proceso de aprendizaje se puede realizar en dos fases: inicialmente se optimizaría la base de reglas y, posteriormente, la base de datos. Por tanto, el AG puede actuar sobre las variables borrosas, las funciones de escalado y sobre las funciones de pertenencia asociadas a dichas variables mediante: el ajuste de parámetros que definen la forma de los conjuntos borrosos, el cruce de conjuntos borrosos con otras variables ó la mutación de puntos que definen los conjuntos borrosos. También puede actuar sobre las reglas lingüísticas de conocimiento, realizando el cruce de antecedentes entre reglas, la mutación de una variable en los antecedentes de la regla o la mutación del conjunto borroso de una variable en una regla.

4.6. Conclusiones

En este capítulo se ha descrito brevemente las características principales de los sistemas borrosos y borrosos-genéticos.

En primer lugar, se han revisado los conceptos básicos de los sistemas borrosos basados en reglas y su aplicación en el control de sistemas mediante controladores borrosos, mostrándose la estructura de estos. Posteriormente, se ha presentado la estructura de un sistema borroso-genético, indicando los principales enfoques que existen en el aprendizaje en dichos sistemas, Pittsburgh, Michigan e IRL. Por último, se ha mostrado con mayor detalle cómo se realiza el aprendizaje evolutivo en el enfoque de Pittsburgh, ya que es uno de los escenarios en los que se aplica el sistema de optimización de parámetros propuesto en esta tesis.

Parte II

Desarrollo de la Investigación

Capítulo 5

Metodología propuesta

5.1. Introducción

A partir de la revisión realizada en los capítulos anteriores se llega a la conclusión de que son necesarias técnicas eficientes que permitan hallar unos "buenos" parámetros con los que se optimice el comportamiento de un AG para un problema dado, ya que, la elección de los mismos determinará en gran medida que el AG encuentre una solución óptima y de forma eficiente.

Tradicionalmente se han utilizado valores elevados para P_s y P_c (si P_s=1.0 se trata de un AG generacional), y valores pequeños para P_m. Los valores que se han considerado más frecuentemente en la literatura son los siguientes:

- N=50, P_s=1.0, P_c=0.6, P_m=0.001 [DeJong, 1975]
- N=30, P_s=0.9, P_c=0.95, P_m=0.01 (evaluación on-line) [Grefenstette, 1986]
- N=80, P_s=0.9, P_c=0.45, P_m=0.01 (evaluación off-line) [Grefenstette, 1986]
- N $\in$ [20,30], P_s=1.0, $P_c \in$ [0.75, 0.95], $P_m \in$ [0.005, 0.01] [Schaffer et al., 1989]
- N=100, P_s=1.0, P_c=0.65, P_m=0.008 [Srinivas y Patnaik, 1994]

Por otra parte, como se ha indicado anteriormente, algunos autores [Hesser y Männer, 1991, Bäck, 1992, Eiben et al., 1999] han sugerido la utilización de técnicas adaptativas, donde la configuración del AG es modificada durante su ejecución, debido a que diferentes valores de los parámetros pueden ser los óptimos en distintos

estados del proceso evolutivo.

En este capítulo se presenta una propuesta para la optimización de los parámetros de un AG. Se trata de un sistema que, con independencia del tipo de codificación utilizado en el algoritmo, tiene como objetivo hallar los "mejores" parámetros en cada tipo de problema, optimizando así su comportamiento. El sistema, además de hallar dichos parámetros, debe encontrar la forma en que variarán a lo largo de la ejecución, para lo cual se ha diseñado un nuevo método de adaptación. En la siguiente sección se muestra la estructura del sistema.

5.2. Sistema de optimización de parámetros

El sistema de optimización propuesto combina dos de las técnicas recogidas en la literatura para mejorar el comportamiento de un AG, como son:

- La meta-evolución, que consiste en buscar el mejor AG para resolver un problema como un problema de optimización y se utiliza otro AG para resolverlo, es decir, existe un meta-AG que opera sobre una población de AGs, donde cada AG es un cromosoma con sus componentes que intentan resolver un problema dado.

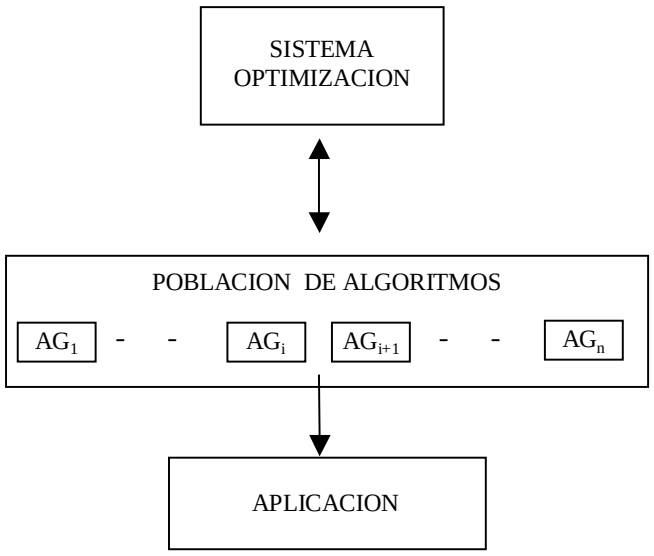

Figura 5.1: *Concepto de meta-evolución.*

- La adaptación de los componentes del AG, que consiste en adaptar o ajustar dinámicamente ciertos componentes durante su ejecución, modificándolos en función de su estado o en función de la información que se tenga sobre el espacio de búsqueda. Como se ha comentado previamente en la sección 2.6, cuando se produce esta adaptación se habla de Algoritmos Genéticos Adaptativos (AGAs).

Por tanto, el sistema de optimización es a su vez un sistema genético, que se ha denominado Sistema Genético Adicional (SGA). Además, como se muestra en la figura 5.2, los AGs de la población pueden ser tanto AGs tradicionales (donde no existe ningún tipo adaptación) como AGAs. Ambos, tanto los AGs tradicionales como los AGAs, pueden a su vez ser generacionales o estacionarios, dependiendo de su configuración. En los AGAs únicamente se adaptan los parámetros de control; en el resto de componentes no se produce ningún tipo de adaptación.

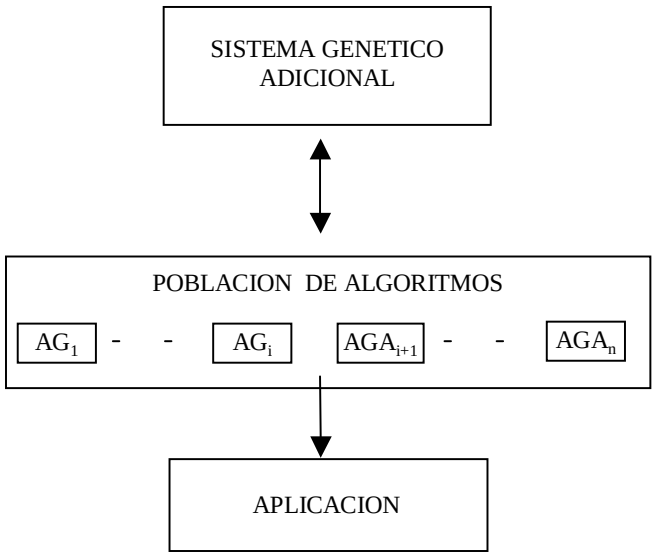

Figura 5.2: *Sistema de optimización propuesto.*

Al tratarse de un sistema genético, el SGA está compuesto por los siguientes componentes:

- Una representación genética de las soluciones para el problema dado.
- Un mecanismo para crear la población inicial de soluciones.
- Una función de evaluación que permita evaluar cada una de las posibles soluciones.
- Un mecanismo de selección de los individuos como padres.
- Un conjunto de operadores genéticos que alteren la composición genética de los nuevos individuos durante la reproducción.
- Los valores de los parámetros que utiliza: N, P_s, P_c y P_m.
- Un mecanismo de sustitución de parte de la población por los nuevos individuos.

A continuación se describen de forma detallada cada uno de estos componentes.

5.2.1. Estructura del sistema

Representación genética

El SGA opera sobre una población de AGs, donde cada individuo puede ser un AG tradicional o un AGA. Sin embargo, dado que el objetivo del SGA es optimizar los parámetros de control, en realidad, cada individuo de la población es un Conjunto de Parámetros (CP_i); el SGA no influye sobre el resto de componentes los cuales se determinan de forma especifica para cada problema.

Por tanto, en la población del SGA, una solución candidata es CP_i, $i = 1, \ldots, n$, que representa a un conjunto de parámetros que se aplican sobre los operadores genéticos de cada AG. Así, cada CP_i es un vector de números reales que contiene los parámetros que se desean optimizar:

$$CP_i = \{p_{i1}, p_{i2}, \ldots, p_{iz}\}$$

donde z es el número máximo de parámetros a optimizar.

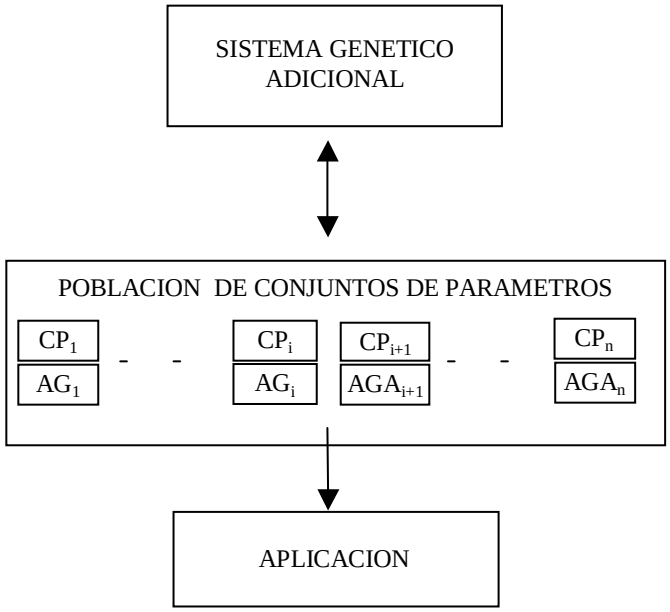

Figura 5.3: *Sistema de optimización de parámetros propuesto.*

Dado que en la literatura no queda muy claro los parámetros que deben optimizarse de forma simultánea, la propuesta es muy flexible en ese sentido y el SGA es fácilmente configurable en cuanto al número de parámetros a optimizar: desde un único parámetro hasta simultáneamente todos ellos. Así, un ejemplo de representación de un individuo podría ser el siguiente:

$$CP_1 = \{N_{1N}, P_{1s}, P_{1c}, P_{1m}\}$$

Como puede apreciarse, con esta representación únicamente pueden ser optimizados parámetros estáticos; sin embargo, se desea que estos parámetros puedan variar a lo largo de la ejecución del algoritmo. Para ello, se propone la utilización de una función $p(t)$ que modifica los parámetros de cada CP_i.

La función propuesta es la siguiente:

$$p(t) = p_0(1 - \frac{(Ln(t+1))^{(1/a)}}{(bLn(T+1))^{(1/a)}}) \quad \begin{matrix} a \in]0,2] \\ b \in [1,\mathrm{T}] \end{matrix}$$

donde p_0 es el valor inicial del parámetro, t es el número de generación o de evaluación en el AG, T es el número máximo de generaciones o evaluaciones en el AG, a es el parámetro que modela la concavidad o convexidad de la curva y b el parámetro que modela la atenuación del valor inicial. La figura 5.4 muestra distintas funciones $p(t)$, en función de los parámetros a y b, para T=500 y $p_0 = 0.75$.

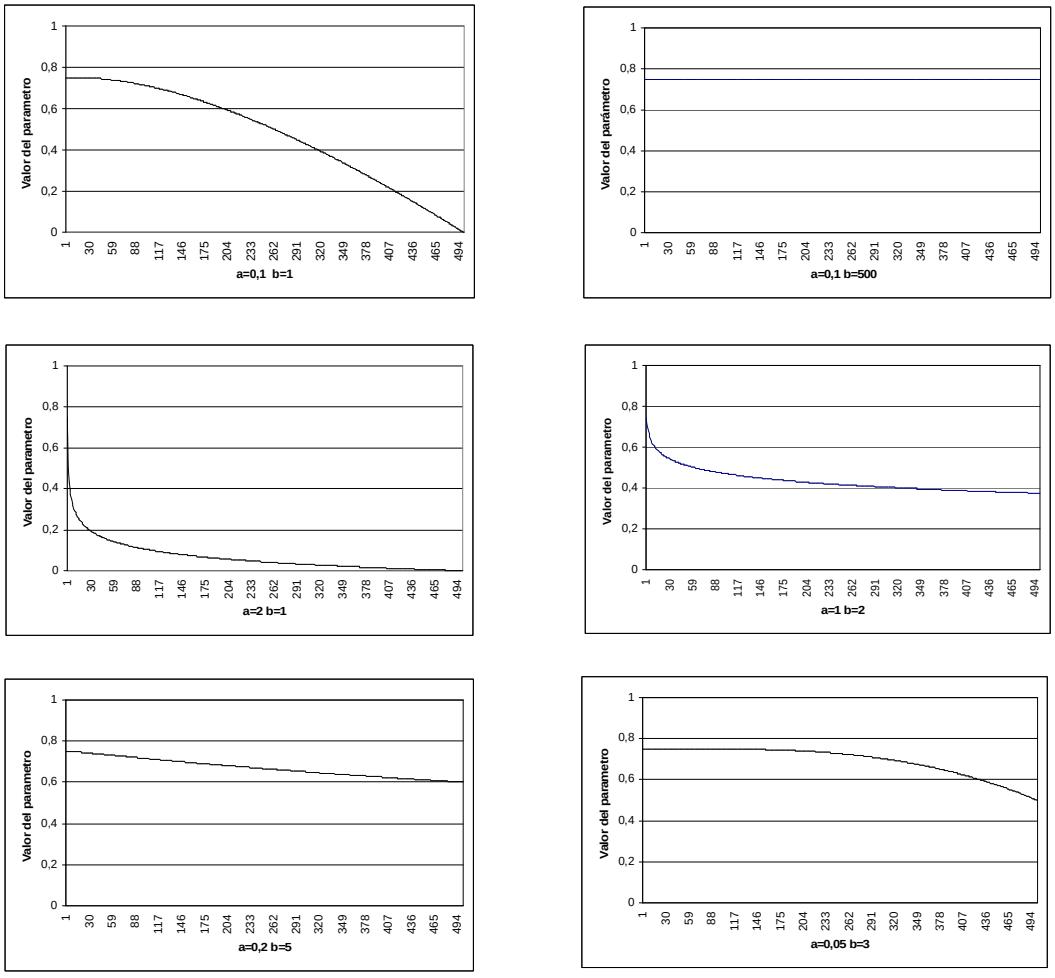

Figura 5.4: *Efecto de la variación de los parámetros a y b.*

Como se observa, la forma en cómo son adaptados los parámetros del AG durante su ejecución depende de los valores de a y b. Para su optimización se introducen dentro de los individuos de la población, es decir, se introducen como parte de los vectores de números reales, CP_i, y evolucionan junto con a ellos, aplicándoles también los operadores de cruce y mutación. Por tanto, la representación genética de los individuos depende del nivel al que se realice la adaptación de los parámetros. En este sentido, el sistema propuesto es también muy flexible y el SGA es fácilmente configurable con los siguientes niveles de adaptación:

- Adaptación a nivel de la población, si se desea que todos los parámetros, de todos los AGs de la población, se adapten de igual forma. Para ello, debe hallarse una única función $p(t)$, que afecta a todos los parámetros de todos los CP_i del mismo modo. En este caso, la representación de los individuos debe ser la siguiente:

  $$CP_i = \{p_{i1}, p_{i2}, \ldots, p_{iz}, a, b\}$$

  como, por ejemplo:

  $$CP_1 = \{N_{1N}, P_{1s}, P_{1c}, P_{1m}, a, b\}$$

- Adaptación a nivel del individuo, si se desea que en cada AG todos los parámetros sean adaptados de igual forma (la adaptación únicamente afectará a ese AG). Para ello, se debe hallar una función $p_i(t)$ para cada CP_i, cuyos parámetros variarán en el tiempo del mismo modo. Para este tipo de adaptación, la representación de cada CP_i debe ser la siguiente:

  $$CP_i = \{p_{i1}, p_{i2}, \ldots, p_{iz}, a_i, b_i\}$$

  como, por ejemplo:

  $$CP_1 = \{N_{1N}, P_{1s}, P_{1c}, P_{1m}, a_1, b_1\}$$

- Adaptación a nivel del componente, si se desea que cada parámetro, en cada AG, sea adaptado de forma independiente. Para que ello sea posible, deben de hallarse distintas funciones $p_{i\lambda}(t)$, $\lambda = 1, \ldots, z$ para cada CP_i, cuyos parámetros variarán en el tiempo de acuerdo con la función que tengan asociada. Para que esto sea posible, la representación de los individuos debe ser la siguiente:

  $$CP_i = \{p_{i1}, p_{i2}, \ldots, p_{iz}, a_{i1}, b_{i1}, a_{i2}, b_{i2}, \ldots, a_{iz}, b_{iz}\}$$

como por ejemplo:

$$CP_1 = \{N_{1N}, P_{1s}, P_{1c}, P_{1m}, a_{11}, b_{11}, a_{12}, b_{12}, \ldots, a_{14}, b_{14}\}$$

y, por tanto, las funciones asociadas a cada parámetro son las siguientes:

$$p_{i\lambda}(t) = p_0^{i,\lambda}(1 - \frac{(Ln(t+1))^{(1/a_{i\lambda})}}{(b_{i\lambda}Ln(T+1))^{(1/a_{i\lambda})}}) \quad \begin{matrix} a_{i\lambda} \in]0,2] \\ b_{i\lambda} \in [1,\mathrm{T}] \end{matrix}$$

La adaptación que se ha estimado más apropiada y se ha implementado finalmente en el SGA ha sido la adaptación a nivel del componente (aunque inicialmente se implementó la adaptación a nivel del individuo), ya que no existe, a priori, razón alguna para que la adaptación sea la misma en todos los parámetros, tanto a nivel del individuo como a nivel de la población. Además, en la adaptación a nivel de la población no es posible disponer, de forma simultánea, de AGs tradicionales y AGAs en la población del SGA.

Por otro lado, se ha considerado no optimizar el parámetro N, tamaño de la población, con el objeto de establecer un punto de referencia a la hora de validar el SGA. De esta forma, en cada escenario de validación, se podrá comparar el comportamiento de un AG al utilizar los parámetros hallados por el SGA con el comportamiento, de ese mismo AG, al utilizar los parámetros recomendados en la literatura para ese tamaño de la población. Evidentemente, con el tipo de representación propuesto, su optimización hubiera sido inmediata. En los escenarios AG binario y AG real, los tamaños de las poblaciones se corresponderán con los utilizados en [Herrera y Lozano, 2003] y en [Baldi et al., 2000] para esos mismos problemas, N=60 y N=50 respectivamente. En el AG híbrido, el tamaño de la población será N=30. Se ha elegido un valor pequeño debido al elevado coste computacional que conlleva la evalución cada base de conocimiento. Por último, indicar que la población de n individuos en el SGA se representa como CP y es un conjunto como el siguiente:

$$CP = (CP_1, \ldots, CP_i \ldots, CP_n)$$

Método para crear la población inicial

Al crear la población inicial, $CP(0)$, se ha considerado conveniente incluir el conocimiento previo que se tiene sobre el problema. En concreto, se ha incluido

como conocimiento previo a un individuo, CP_1, cuyos parámetros se corresponden con los recomendados en la literatura. El resto de la población inicial, $CP_2, \ldots, CP_n$ se crea de forma aleatoria, generando individuos al azar, con probabilidad uniforme, sobre el espacio de búsqueda. Los parámetros están comprendidos en los siguientes intervalos:

- $P_s, P_c, P_m \in [0, 1]$
- $a_{i\lambda} \in]0, 2]$
- $b_{i\lambda} \in [1, T]$. Si el número de generaciones o evaluaciones en el AG es muy elevado, puede acotarse el valor $b_{i\lambda}$ en el intervalo [1,100], ya que, para T=100 puede considerarse que la atenuación es práctica nula.

Función de evaluación

Para evaluar los individuos CP_i de la población CP, se propone realizar una evaluación basada en la media después de T generaciones, es decir, una evaluación que mide el comportamiento del AG a través del comportamiento medio de los individuos obtenidos al final de la ejecución del mismo. Así, si la población de individuos en la generación t de un AG se denota como $P(t) = \{x_1^t, x_2^t, \ldots, x_N^t\}$ y cada x_α^t es evaluado por una función f, $f(x_\alpha^T)$ proporciona la evaluación del individuo x_α al final de la ejecución del AG. Por tanto, la evaluación de CP_i después de T generaciones se obtiene de la siguiente forma:

$$f(CP_i) = \frac{\sum_{\alpha=1}^{N} f(x_\alpha^T)}{N}$$

Método de selección

El mecanismo de selección es el encargado de seleccionar los individuos CP_i que van a reproducirse, es decir, los individuos que van a actuar como padres. El mecanismo debe garantizar que los mejores individuos, los más aptos, tengan un mayor número de oportunidades de reproducción frente a los menos aptos, por lo que la selección de un CP_i está directamente relacionado con su valor de aptitud, $f(CP_i)$. El método de selección propuesto se basa en una selección determinística, haciendo uso de una presión selectiva muy elevada y seleccionándose como padres aquellos μ individuos (dados por la P_s del SGA) que obtengan una mayor aptitud. Por tanto, con la utilización de este método se trata de conseguir una convergencia elevada.

Operadores genéticos

Los operadores genéticos están basados en el cruce y la mutación de la información genética de los individuos seleccionados de la población CP. El operador de cruce permite mezclar la información genética de los padres escogidos, con objeto de obtener la nueva descendencia. Una vez generados los nuevos individuos, tras aplicar el operador de cruce, el operador de mutación podrá realizar alteraciones aleatorias sobre la información genética de los mismos. Con los operadores que se proponen se trata de obtener una alta diversidad en la población, teniendo en cuenta que la representación de los individuos de la población CP es de tipo real.

Operador de cruce.

El funcionamiento del operador de cruce propuesto es el siguiente:

dados dos individuos de la población, CP_j y CP_k que se van a cruzar,

$$CP_j = \{p_{j1}, \ldots, p_{jz}, a_{j1}, b_{j1}, \ldots, a_{jz}, b_{jz}\} \quad \begin{array}{l} p_{j\lambda} \in [0,1] \\ a_{j\lambda} \in]0,2] \\ b_{j\lambda} \in [1,\mathrm{T}] \end{array}$$

$$CP_k = \{p_{k1}, \ldots, p_{kz}, a_{k1}, b_{k1}, \ldots, a_{kz}, b_{kz}\} \quad \begin{array}{l} p_{k\lambda} \in [0,1] \\ a_{k\lambda} \in]0,2] \\ b_{k\lambda} \in [1,\mathrm{T}] \end{array}$$

se divide el intervalo de actuación de los parámetros, p_λ, a_λ y b_λ, en tres sub-intervalos para obtener los descendientes, CP_j' y CP_k'. Estos 3 sub-intervalos pueden ser clasificados como sub-intervalos de exploración o explotación, como puede apreciarse en las figuras 5.5, 5.6 y 5.7.

El intervalo de actuación de los parámetros p_λ, $[0,1]$, se divide en las regiones $[0, P_{min}]$, $[P_{min}, P_{max}]$ y $[P_{max}, 1]$, siendo $P_{min} = min(p_{j\lambda}, p_{k\lambda})$ y $P_{max} = max(p_{j\lambda}, p_{k\lambda})$. En cuanto al intervalo de actuación de los parámetros a_λ, $]0,2]$, este se divide en las regiones $[0, A_{min}]$, $[A_{min}, A_{max}]$ y $[A_{max}, 2]$, siendo $A_{min} = min(a_{j\lambda}, a_{k\lambda})$

y $A_{max} = max(a_{j\lambda}, a_{k\lambda})$. Por último, el intervalo de actuación de los parámetros b_λ, $[0, T]$, se divide en las regiones $[0, B_{min}]$, $[B_{min}, B_{max}]$, y $[B_{max}, T]$, siendo $B_{min} = min(b_{j\lambda}, b_{k\lambda})$ y $B_{max} = max(b_{j\lambda}, b_{k\lambda})$.

Así, la descendencia será la siguiente:

$$CP_j^{'} = \{p_{j1}^{'}, \ldots, p_{jz}^{'}, a_{j1}^{'}, b_{j1}^{'}, \ldots, a_{jz}^{'}, b_{jz}^{'}\}$$

$$CP_k^{'} = \{p_{k1}^{'}, \ldots, p_{kz}^{'}, a_{k1}^{'}, b_{k1}^{'}, \ldots, a_{kz}^{'}, b_{kz}^{'}\}$$

donde

$$p_{j\lambda}^{'} = \frac{p_{j\lambda} + p_{k\lambda}}{2}$$

$$p_{k\lambda}^{'} = \begin{cases} \frac{P_{min}}{2} & \alpha < \frac{1}{2} \\ \frac{1-P_{max}}{2} & \alpha > \frac{1}{2} \end{cases}$$

siendo α un valor aleatorio en el intervalo [0,1]. De este modo se permite que uno de los descendientes esté localizado fuera del intervalo $[P_{min}, P_{max}]$, con el objeto de proporcionar diversidad en la población y prevenir la convergencia prematura.

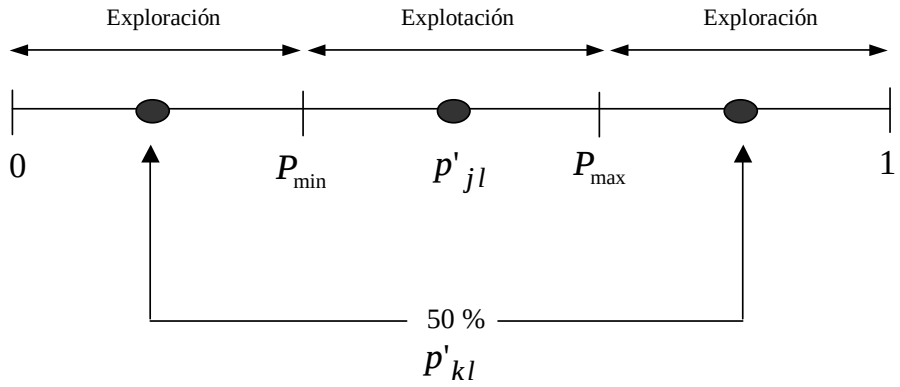

Figura 5.5: *Operador de cruce propuesto para los parámetros p_λ.*

Los parámetros $a'_{j\lambda}$, $a'_{k\lambda}$, $b'_{j\lambda}$ y $b'_{k\lambda}$ serán los siguientes:

$$a'_{j\lambda} = \frac{a_{j\lambda} + a_{k\lambda}}{2}$$

$$b'_{j\lambda} = \frac{b_{j\lambda} + b_{k\lambda}}{2}$$

situados en la zona de explotación de la información y,

$$a'_{k\lambda} = \begin{cases} \frac{A_{min}}{2} & \alpha_1 < \frac{1}{2} \\ \frac{2 - A_{max}}{2} & \alpha_1 > \frac{1}{2} \end{cases}$$

$$b'_{k\lambda} = \begin{cases} \frac{B_{min}}{2} & \alpha_2 < \frac{1}{2} \\ \frac{T - B_{max}}{2} & \alpha_2 > \frac{1}{2} \end{cases}$$

situados en la zona de exploración, donde α_1 y α_2 son valores aleatorios en el intervalo [0,1].

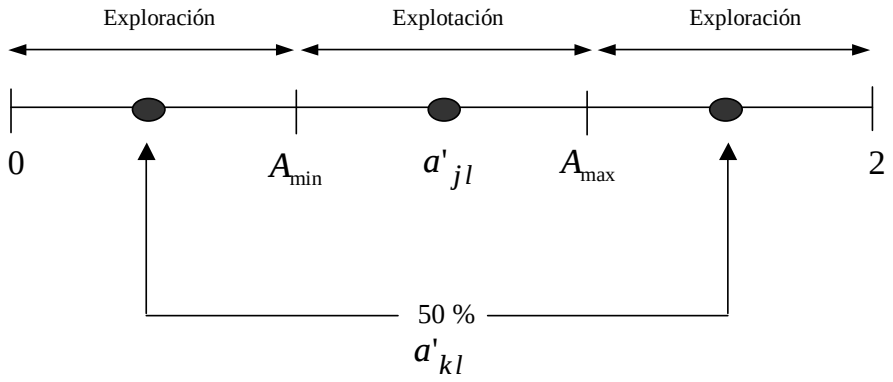

Figura 5.6: *Operador de cruce propuesto para los parámetros a_λ.*

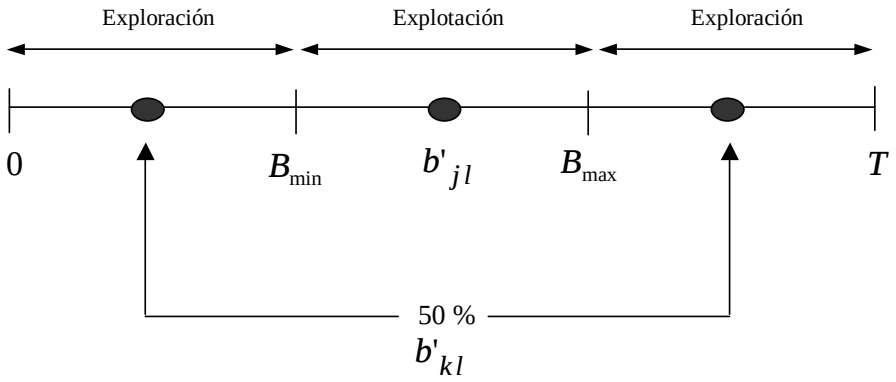

Figura 5.7: *Operador de cruce propuesto parámetros* b_λ.

Operador de mutación.

Una vez generados los nuevos individuos tras aplicar el operador de cruce, el operador de mutación puede realizar alteraciones aleatorias sobre la información genética de dichos individuos. Es importante que este operador permita alcanzar cualquier parte del espacio de búsqueda. Para ello se utiliza el operador random mutation -mutación aleatoria-, donde, para cada uno de los genes del individuo, la mutación consiste en sustituir, con una probabilidad dada, los valores de dichos genes por otros generados aleatoriamente con distribución uniforme en el intervalo de sus respectivos dominios. Por tanto, tras aplicar el operador de mutación a los individuos CP'_j y CP'_k se obtendrán los nuevos individuos:

$$CP''_j = \{p''_{j1}, \ldots, p''_{jz}, a''_{j1}, b''_{j1}, \ldots, a''_{jz}, b''_{jz}\}$$

$$CP''_k = \{p''_{k1}, \ldots, p''_{kz}, a''_{k1}, b''_{k1}, \ldots, a''_{kz}, b''_{kz}\}$$

Valores de los parámetros de control

Como se ha comentado en el capítulo 2, un aspecto muy importante dentro de la estructura de un AG son los valores de los parámetros que utiliza. Posiblemente, en la elección de los parámetros del SGA lo más sencillo hubiera sido aplicar un conjunto

de valores estándar recogidos en la literatura. Sin embargo, y aún siguiendo las recomendaciones de distintos autores, se han introducido algunas variaciones. Así, los valores propuestos para el SGA son los siguientes:

- En cuanto al tamaño de la población, n, se ha seleccionado uno de los menores valores mostrados en la literatura, en concreto, n=20 [Schaffer et al., 1989]. Esto se debe, fundamentalmente, al elevado coste computacional que conlleva una población grande en el SGA.

- La probabilidad de selección que se ha elegido es P_s=0.3, por tanto, se escogen como padres aquellos seis CP_i que obtengan una mayor aptitud. De esta forma, el SGA se corresponde con un modelo estacionario. A priori, puede considerarse un valor pequeño; sin embargo, existen modelos de AGs, con los que se han obtenido buenos resultados, en los que en cada iteración se seleccionan únicamente dos padres y, tras aplicarles los operadores genéticos, la descendencia reemplaza a uno ó dos individuos de la población. Siguiendo esta línea de actuación, se ha considerado aplicar una P_s pequeña, optando por el valor de 0.3 al ser el menor de los posibles valores que podía tomar P_s en los experimentos realizados por Grefenstette [Grefenstette, 1986].

- En cuanto a la probabilidad de cruce, P_c, se propone un valor elevado, en concreto P_c=1.0. Por tanto, siempre se producirá el cruce de los padres seleccionados.

- Para la probabilidad de mutación, P_m, se propone un valor pequeño, en concreto P_m=0.05. Aún siendo pequeño, es uno de los mayores valores encontrados en la literatura [Srinivas y Patnaik, 1994].

Método de sustitución

Al tratarse el SGA de un modelo estacionario, en cada iteración se reemplaza solamente una parte de la población CP. Por tanto, se hace necesario un método de sustitución a través del cual los individuos de la población sean reemplazados por los nuevos descendientes; teniendo en cuenta que para insertar un nuevo individuo debe de eliminarse, previamente, otro de la población. Dado que la presión selectiva también se ve afectada por el método de sustitución, se propone la utilización del

método Replace Worst Strategy -reemplazo de los peores- que proporciona una presión selectiva elevada, reemplazándose los peores CP_i de la población. Por tanto, se deduce que el SGA es a su vez elista.

Con la elección de estos operadores en el SGA se ha tratado de proporcionar al algoritmo una alta convergencia, a través de los métodos de selección y sustitución, que proporcionan una presión selectiva elevada, y una alta diversidad, a través de los operadores de cruce y mutación. En la figura 5.8 se muestra el procedimiento general de funcionamiento.

Por otro lado, nos gustaría indicar que los parámetros del SGA podrían ser optimizados, a su vez, mediante un meta-meta-AG. Sin embargo, consideramos que la mejora que ello supondría sobre los resultados en los sistemas finales no compensaría el enorme coste computacional que ello conllevaría.

Los siguientes capítulos, 6, 7 y 8, describen los escenarios donde ha sido aplicado el sistema de optimización propuesto, los experimentos realizados así como los resultados obtenidos en cada uno de ellos. En todos los casos, la metodología seguida ha sido la siguiente:

- En primer lugar, en cada uno de los escenarios, se ha hallado, con el sistema propuesto, los valores de los parámetros así como la forma en que pueden variar durante la ejecución del AG.

- Posteriormente, se ha comprobado el comportamiento del AG con dichos valores. A este algoritmo se le ha denominado AG_SGA.

- Seguidamente, se han comparado los resultados obtenidos con AG_SGA con: a) los obtenidos al utilizar los parámetros estáticos recomendados en la literatura y, b) los obtenidos al utilizar los principales métodos de adaptación de las probabilidades de mutación. Se ha considerado la adaptación de este único parámetro (manteniendo el resto con valores estáticos estándares) porque es el que puede determinar directamente el grado de diversidad en la población [Herrera y Lozano, 2003, Rojas et al., 2002], que es el principal factor para evitar el problema de la convergencia prematura.

Procedimiento Sistema Genético Adicional

Inicio_Sistema_Genético_Adicional

it_sga=0;

Obtención de la población inicial CP(it_sga);

Mientras (no se cumplan los criterios de finalización) **Hacer**

Para cada CP_i (i=1, ..., n)

Inicio_Algoritmo_Genetico

- t=0;
- Obtención de la población inicial P(t);
- Evaluación de la población P(t);
- **Mientras** (no se cumplan los criterios de finalización) **Hacer**
  t=t+1;
  Selección de una nueva P(t) a partir de P(t-1);
  Aplicar operadores de cruce y mutación sobre P(t);
  Sustitución de parte de P(t-1) por la descendencia;
  Evaluación de P(t);

Final_Algoritmo_Genetico

Evaluación de CP_i;

it_sga=it_sga+1;

Selección de una nueva población CP(it_sga) a partir de CP(it_sga-1);

Aplicar operadores de cruce y mutación sobre CP(it_sga);

Sustitución de parte de CP(it_sga-1) por la descendencia;

Final_Sistema_Genetico_Adicional

Figura 5.8: *Estructura del Sistema Genético Adicional.*

- Por último, para llevar a cabo la comparación de los resultados, se han ejecutado todos los algoritmos 30 veces y, a continuación, se ha realizado un test estadístico. En concreto, y siempre que ha sido posible, se ha aplicado el t-Student que es el test que se utiliza para el contraste de hipótesis sobre medias en poblaciones. Para determinar si existen diferencias significativas en dichas medias se ha utilizado el valor estándar de p-valor <0.05. Se ha denominado hipótesis nula (H_0) a la hipótesis que se supone cierta de partida e hipótesis alternativa (H_1) a la que reemplazará a la hipótesis nula cuando ésta sea rechazada. La hipótesis que contrasta el test estadístico es:

  - $H_0 : \mu_1 = \mu_2$; Las medias son iguales
  - $H_1 : \mu_1 \neq \mu_2$; Las medias son diferentes

  Las hipótesis previas que se deben cumplir para poder aplicar el t-Student son:

  a) que en cada grupo de datos, la variable bajo estudio siga una distribución normal y,

  b) que la dispersión en ambos grupos de datos sea homogénea, lo que se conoce como hipótesis de homocedasticidad (igualdad de varianzas).

  La prueba estadística más utilizada para contrastar la normalidad de los datos es la de Kolmogorov-Smirnov mientras que para contrastar la homogeneidad de las varianzas es la prueba de Levene. Para determinar la significación en estas pruebas también se ha utilizado el valor estándar de p-valor <0.05. Cuando no se ha cumplido la hipótesis de normalidad se ha realizado una transformación de los datos para tratar de normalizarlos. En este sentido, se ha utilizado la transformación logaritmo neperiano, que es una de las más utilizadas. Si aún así no se ha podido asumir la hipótesis de normalidad, se ha sustituido el t-Student por la prueba no paramétrica de Mann-Whitney que es una prueba alterna a la t-Student para comparar dos muestras independientes. Cuando ha existido normalidad en los datos y no se ha cumplido la hipótesis de homocedasticidad se ha utilizado una modificación de la t-Student, que es válida para el caso de no homogeneidad de varianzas. Para la realización de los distintos tests se ha utilizado el software de análisis estadístico SPSS.

5.3. Conclusiones

En este capítulo se ha presentado una propuesta de una nueva técnica de optimización de parámetros en un AG, la cual presenta las siguientes caracteristicas:

- Combina dos de los procedimientos recogidos en la literatura para mejorar el comportamiento de un AG, como son la metaevolución y la adaptación de los parámetros en el AG.
- Puede ser utilizada con independencia del tipo de codificación existente en el AG.
- Es independiente del problema específico que el AG trate de resolver.
- Es flexible en relación al número de parámetros de se deseen optimizar.

Capítulo 6

AG binario. Optimización de funciones

6.1. Introducción

En este capítulo se describe el primero de los tres escenarios donde ha sido aplicado el sistema de optimización propuesto, así como los experimentos realizados y los resultados obtenidos. Se trata de un AG clásico, con codificación binaria, para la resolución de un problema de minimización sobre un conjunto de seis funciones representativas, utilizadas frecuentemente en la literatura como funciones de evaluación del comportamiento de un AG. En la tabla 6.1 se indican dichas funciones, cuyas caracteristicas se muestran con mayor detalle en el apartado 6.4. El objetivo perseguido por el SGA es hallar, para cada una de las funciones, un conjunto de parámetros que optimicen, en cada caso, el comportamiento del AG.

Tabla 6.1: *Funciones de prueba utilizadas.*

Función	**Evaluación del individuo óptimo**
Función Sphere Model (f_{Sph})	0
Función de Rosenbrock (f_{Ros})	0
Función de Rastrigin (f_{Ras})	0
Función One-max (f_{One})	0
Función Fully deceptive order-3 (f_{Dec})	0
Función Royal Road (f_{RR})	0

A continuación, se describe la estructura del AG clásico empleado y, posteriormente, en el apartado 6.3 se muestra cómo se ha realizado la integración del SGA con dicho AG.

6.2. Estructura del AG clásico

Para la implementación del AG clásico de esta parte de la tesis doctoral se ha utilizado la librería GAlib, del Massachusetts Institute of Technology [GAlib]. GAlib es un conjunto de objetos implementados en C++ que pueden ser integrados perfectamente en cualquier programa realizado en C++.

Los componentes que se han definido para el AG son los siguientes:

1. Representación genética de las soluciones. Las variables de las funciones f_{Sph}, f_{Ros} y f_{Ras} han sido codificadas de forma binaria (*bit string*) utilizando código Gray y con 20 genes cada una de ellas. En el resto de las funciones, f_{One}, f_{Dec}, f_{RR}, los individuos también se han codificado como *bit string*, pero en este caso con codificación binaria natural, con una longitud de 400, 39 y 200 bits respectivamente, como utilizan los autores en [Herrera y Lozano, 2003].

2. Método de formación de la población inicial. La población inicial se crea de forma aleatoria, generando individuos al azar.

3. Sistema de evaluación, que evalúa el grado de adaptación de cada individuo proporcionando un valor de aptitud. En el caso de las funciones f_{Sph}, f_{Ros} y f_{Ras}, la evaluación de los individuos se realiza a través de la función objetivo. Debido a la utilización del código Gray, las evaluaciones de los individuos óptimos son 2.4e-10, 1.5e-09 y 4.8e-08 respectivamente. En el resto de las funciones, al no utilizarse el código Gray, la evaluación de los óptimos será 0.

   Para determinar la evaluación en la función f_{One}, el valor de la función objetivo se resta a 400 (que es el valor máximo) con el fin de asignar una evaluación cero al individuo óptimo.

   $$f_{Eval\_One}(\vec{x}) = 400 - f_{One}(\vec{x})$$

   En el caso de la función f_{Dec}, el problema es considerado como una concatenación de 13 subproblemas de longitud 3. Para determinar la evaluación de

los individuos, el valor de la función objetivo se resta a 390 (que es el valor máximo) con el fin de también asignar una evaluación de cero al individuo óptimo.

$$f_{Eval\_Dec}(x) = 390 - f_{Dec}(x)$$

Por último, para determinar la evaluación de los individuos en la función f_{RR}, el valor de la función se resta a 200 (que es el valor máximo).

$$f_{Eval\_RR}(x) = 200 - f_{RR}(x)$$

4. Método de selección. El mecanismo de selección permite elegir a un porcentaje de individuos de la población, dado por P_s, que van a disponer de oportunidades de reproducirse. Se ha empleado el método Linear Ranking Selection -Orden Lineal- para el cálculo de las probabilidades de reproducción, donde los individuos son ordenados en función de su bondad y se asigna una probabilidad de reproducción a cada uno dependiendo de su orden. Por otro lado, se ha utilizado el algoritmo de muestreo Stochastic Universal Sampling -Muestreo Universal Estocástico- para la selección de los individuos.

5. Operadores genéticos. Durante la fase de reproducción se han utilizado los siguientes operadores:

   - Cruce. El cruce de los individuos seleccionados como padres se ha realizado mediante el operador de cruce Single Point Crossover (SPX) -cruce de 1 punto-.
   - Mutación. Con este operador se alteran aleatoriamente los componentes de los individuos descendientes obtenidos tras el proceso de cruce. El operador empleado ha sido el denominado Clock Mutation -mutación de reloj-.

6. Valores de los parámetros de control. El tamaño de la población es de 60 individuos, que es el utilizado en [Herrera y Lozano, 2003] para este mismo problema. El resto de valores de los parámetros se corresponden con los de cada individuo CP_i del SGA. Inicialmente son valores aleatorios (a excepción de CP_1) y evolucionan hacia valores más óptimos al aplicar el SGA propuesto.

7. Método de sustitución. Una vez generados los nuevos individuos, tienen que ser introducidos en la población. Para que el número de individuos permanezca

constante hay que eliminar a tantos como los que se van a introducir. La sustitución se realiza mediante el método Replace Worst Strategy (RW) -reemplazo de los peores-, donde se reemplazan los peores individuos de la población. En los AGs generacionales (P_s=1.0) se ha utilizado una estrategia elitista, donde no se reemplaza al mejor individuo de la población en cada iteración.

6.3. Integración con el sistema de optimización

Con el objeto de hallar los parámetros que optimizan el comportamiento del AG en cada una de las funciones se ha integrado el SGA con el AG clásico, como se muestra en la figura 6.1.

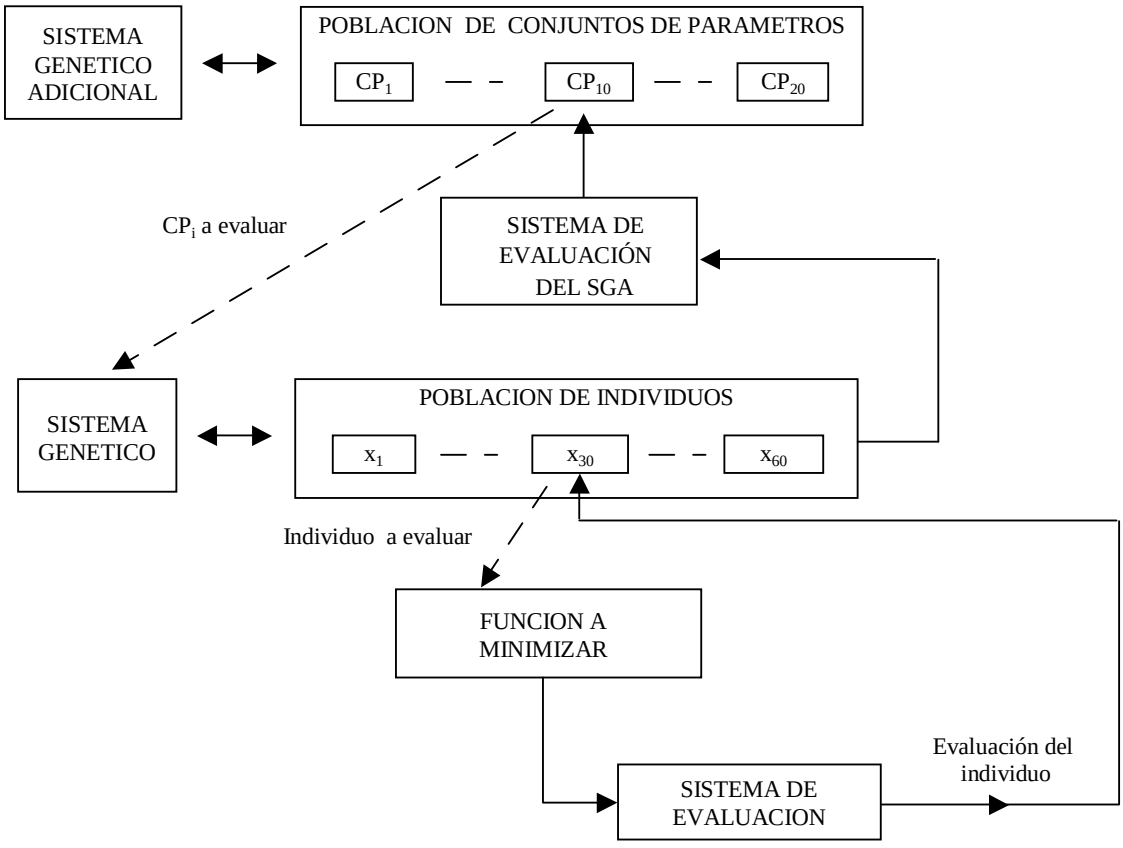

Figura 6.1: *Aprendizaje evolutivo del Conjunto de Parámetros en un AG binario.*

En concreto, los parámetros a optimizar son P_s, P_c y P_m, por lo que la representación, en este escenario, de cada CP_i es la siguiente:

$$CP_i = \{P_{is}, P_{ic}, P_{im}, a_{i1}, b_{i1}, a_{i2}, b_{i2}, a_{i3}, b_{i3}\}$$

De esta forma, la variación de cada parámetro en cada CP_i durante la ejecución del AG viene dada por:

$$P_{is}(t) = p_0^{is}(1 - \frac{(Ln(t+1))^{(1/a_{i1})}}{(b_{i1}Ln(T+1))^{(1/a_{i1})}}) \quad \begin{matrix} a_{i1} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{ic}(t) = p_0^{ic}(1 - \frac{(Ln(t+1))^{(1/a_{i2})}}{(b_{i2}Ln(T+1))^{(1/a_{i2})}}) \quad \begin{matrix} a_{i2} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{im}(t) = p_0^{im}(1 - \frac{(Ln(t+1))^{(1/a_{i3})}}{(b_{i3}Ln(T+1))^{(1/a_{i3})}}) \quad \begin{matrix} a_{i3} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

Así, el proceso de aprendizaje de los parámetros del AG así como de los parámetros $a_{i\lambda}$ y $b_{i\lambda}$ de cada función $p_{i\lambda}(t)$ se realiza de la siguiente forma:

En primer lugar, se parte de una población inicial de Conjuntos de Parámetros, CP(0), compuesta por 20 individuos CP_i.

Posteriormente, mientras que el número de iteraciones en el SGA (it_sga) sea inferior a 500:

- Se evalúa la población CP(it_sga).

  Para evaluar cada CP_i de la población se ejecuta un AG, que utiliza los valores de los parámetros que contiene dicho CP_i. El AG parte de una población inicial P(0) y tras realizarse 50000 evaluaciones se obtiene una nueva población, P(50000). A partir de los individuos obtenidos en P(50000) se evalúa CP_i.

- Una vez evaluados todos los individuos de CP(it_sga), se incrementa el número de iteraciones en el SGA (it_sga=it_sga+1).

- Seguidamente, se selecciona una nueva población CP(it_sga) a partir de CP (it_sga-1).

- A continuación, se aplican los operadores de cruce y mutación sobre CP(it_sga).

- Por último, se sustituye parte de CP(it_sga-1) por la descendencia generada.

Al final de la ejecución del Sistema Genético Adicional se obtiene una población diferente a la inicial, CP(500), y el CP_i cuya evaluación es la menor, es el que contiene los mejores valores.

6.4. Funciones de prueba

Las funciones de prueba seleccionadas presentan distintas propiedades y son utilizadas frecuentemente en la literatura como funciones de evaluación del comportamiento de un AG. Las seis funciones que se han considerado son las siguientes:

- *Función Sphere Model (f_{Sph})*, también conocida como función nº 1 de De Jong [DeJong, 1975]. Con el fin de establecer un criterio estándar con el que evaluar la efectividad de un AG, De Jong propuso una familia de funciones conocidas con el nombre de funciones De Jong. Estas funciones pueden ser continuas o discretas, unimodales o multimodales, cóncavas o convexas, cuadráticas o no cuadráticas, etc. En el caso que nos ocupa, la función se define como:

$$f_{Sph}(\vec{x}) = \sum_{i=1}^{n} x_i^2$$

  donde $n = 10$ y -5.12 $\leq x_i \leq$ 5.12. La evaluación del individuo óptimo es $f_{Sph}(x^*) = 0$. Esta función es un ejemplo representativo de una función continua, convexa y unimodal. En la figura 6.2 se muestra una vista para n=2.

- *Función Rosenbrock (f_{Ros})*, también conocida como función nº 2 de De Jong [DeJong, 1975]. Se trata de una función continua y unimodal, que presenta un estrecho valle entre los mínimos locales y el mínimo global. Como puede observarse en la figura 6.3, el mínimo global se encuentra dentro del largo y estrecho valle. La función se define como:

$$f_{Ros}(\vec{x}) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$$

  donde $n = 2$ y -5.12 $\leq x_i \leq$ 5.12. La evaluación del individuo óptimo es $f_{Ros}(x^*) = 0$.

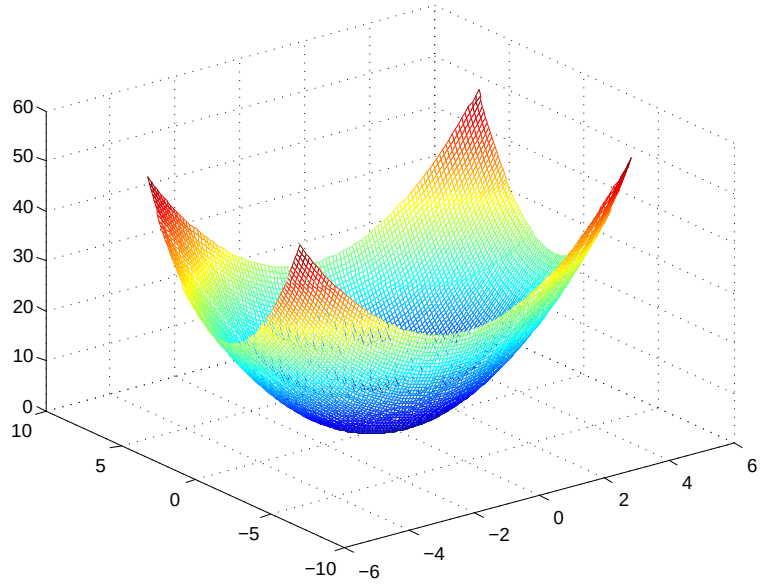

Figura 6.2: *Función Sphere Model con n=2.*

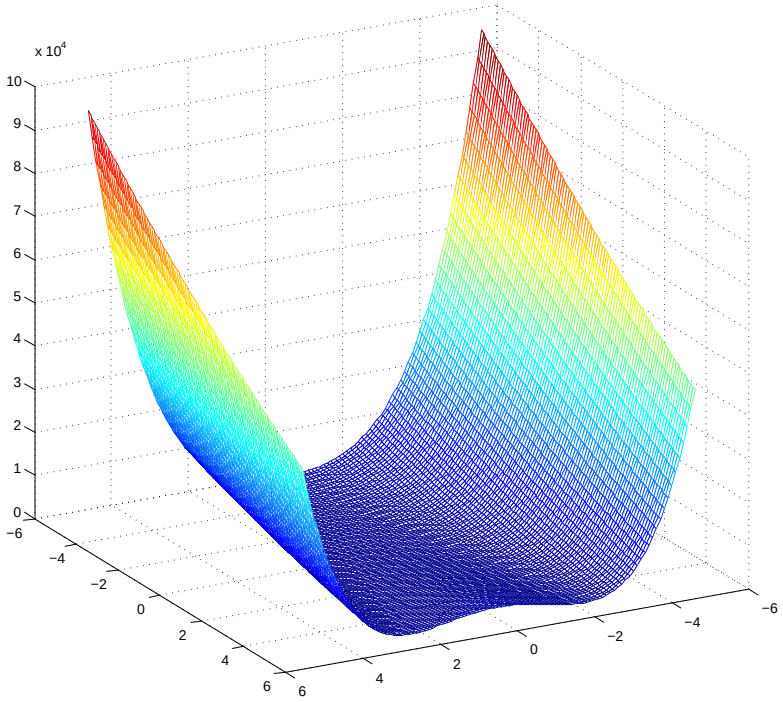

Figura 6.3: *Función Rosenbrock con n=2.*

- *Función Rastrigin (f_{Ras})* [Bäck, 1991]. Representa un ejemplo típico de función multimodal, no lineal y continua, donde encontrar el mínimo global es bastante complicado debido al gran número de mínimos locales existentes, como puede apreciarse en la figura 6.4. La función se define como:

$$f_{Ras}(\vec{x}) = a * n + \sum_{i=1}^{n} x_i^2 - a * cos(w * x_i)$$

donde n=10, a=10, w=2*π y -5.12 $\leq x_i \leq$ 5.12. La evaluación del individuo óptimo es $f_{Ras}(x^*) = 0$. El entorno cercano al mínimo global se muestra en la figura 6.5.

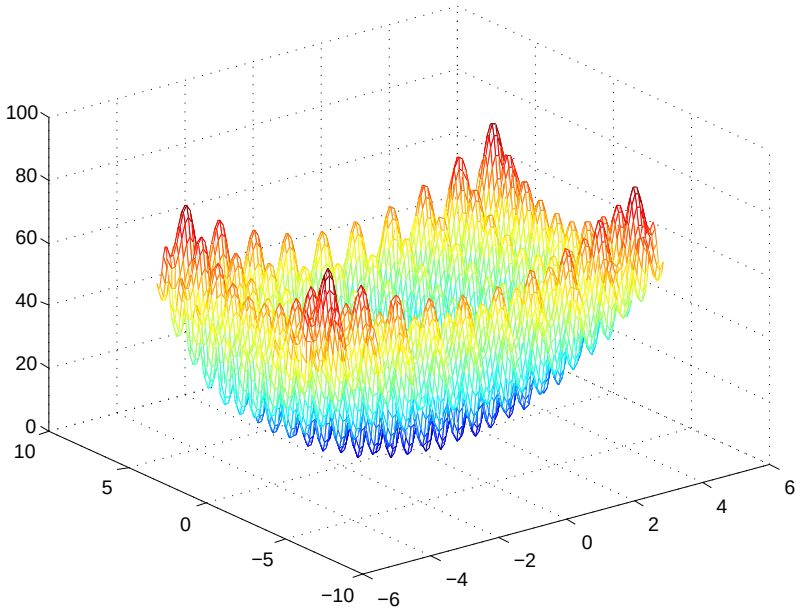

Figura 6.4: *Función Rastrigin con n=2.*

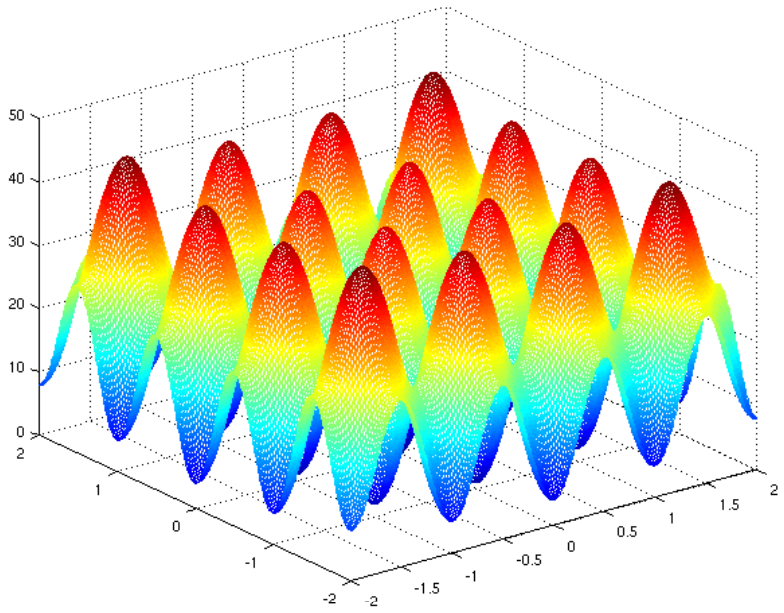

Figura 6.5: *Función Rastrigin con n=2 cerca del mínimo global.*

- *Función One-Max (f_{One}).* Esta función devuelve un número, que se corresponde con el número de 1's que contiene el individuo, que es un *bit string* de 400 bits. Por tanto, el individuo que maximiza esta función es aquel en el que todos sus bits son 1, coincidiendo, en ese caso, el valor de la función con la longitud del individuo. La función es la siguiente:

$$f_{One}(\vec{x}) = \sum_{i=1}^{400} x_i$$

  donde x_i puede ser 1 ó 0. Dado que en nuestro caso se trata de un problema de minimización, para determinar la evaluación de cada individuo el valor de esta función se resta a 400, siendo la evaluación del individuo óptimo $f_{Eval\_One}(x^*) = 0$.

- *Función Fully deceptive order-3 (f_{Dec})* [Goldberg et al., 1989].

  En este tipo de problemas existen ciertos esquemas genéticos que dirigen la búsqueda hacia una solución que no es la óptima. El algoritmo es engañado, creyendo que el proceso de optimización va bien; sin embargo, existe información genética que lo dirige hacia una solución errónea. Esto se debe a que

los esquemas genéticos que contienen el óptimo global no tienen la suficiente importancia y no proliferan durante el proceso genético. En nuestro caso, el problema consiste en la concatenación de 13 subproblemas de 3 bits, tratándose por tanto de un problema de 39 bits. La evaluación para cada secuencia de 3 bits del *bit string* es la mostrada en la tabla 6.2. Por tanto, la función proporciona, para cada individuo, un valor que es la suma de las evaluaciones de las secuencias de 3 bits de cada subproblema, siendo el valor máximo 390.

Tabla 6.2: *Evaluación en la función Fully deceptive order-3.*

Cromosomas de 3 bits	**Evaluación**
000	28
001	26
010	22
100	14
110	0
011	0
101	0
111	30

- *Función Royal Road (f_{RR}).* En 1993, Forrest y Mitchell [Forrest y Mitchell, 1993] idearon unas funciones denominadas *Royal Road functions* donde los AGs encuentran bastantes dificultades en encontrar el óptimo. En nuestro caso se trata de un problema de 200 bits, compuesto por 25 bloques contiguos de ocho bits. Cada bloque es evaluado con una puntuación de 8 si todos los bits son 1's. Por tanto, el valor máximo que puede retornar la función es de 200.

6.5. Resultados

Para evaluar el comportamiento del sistema de optimización de parámetros propuesto se han realizado numerosos experimentos en este escenario, fruto de los cuales se han obtenido diversos resultados que son analizados en esta sección.

El objetivo ha sido encontrar los parámetros que optimizan el comportamiento del AG en cada una de las funciones, los cuales pueden variar su valor durante la ejecución del mismo. Para hallar dichos parámetros así como los que hacen variar su valor, a_i y b_i, mediante cada una de las funciones $p_i(t)$, se ha añadido el Sistema Genético Adicional. En este caso se han utilizado tres funciones $p_i(t)$ i =1, 2, 3 que han afectado, de forma independiente, a los parámetros P_s, P_c y P_m respectivamente. Cada AG se ha ejecutado con un máximo de 50000 evaluaciones mientras que en el Sistema Genético Adicional se han ejecutado 500 iteraciones. Al final del experimento se ha obtenido una población diferente a la inicial en el Sistema Genético Adicional y se ha seleccionado el CP_i cuya puntuación ha sido menor.

6.5.1. Parámetros hallados con el sistema propuesto

La tablas 6.3 y 6.4 muestran los valores del mejor CP_i obtenido para cada una de las funciones.

Tabla 6.3: *Valores de los parámetros hallados en las funciones f_{Sph}, f_{Ros} y f_{Ras}.*

Parámetro	**Valor en f_{Sph}**	**Valor en f_{Ros}**	**Valor en f_{Ras}**
P_s	0.550	0.200	0.097
P_c	0.934	0.420	0.796
P_m	0.017	0.025	0.161
a_1 - b_1	1.60 - 2.50	1.90 - 36.27	1.25 - 19.94
a_2 - b_2	0.07 - 22.29	1.05 - 3.50	1.90 - 31.01
a_3 - b_3	0.62 - 1.00	1.61 - 28.50	0.91 - 1.00

Tabla 6.4: *Valores de los parámetros hallados en las funciones f_{One}, f_{Dec} y f_{RR}.*

Parámetro	**Valor en f_{One}**	**Valor en f_{Dec}**	**Valor en f_{RR}**
P_s	0.640	1.0	0.929
P_c	0.800	0.357	0.138
P_m	0.0027	0.02503	0.0032
a_1 - b_1	0.25 - 45.00	0.80 - 8.31	0.77 - 24.44
a_2 - b_2	1.00 - 15.50	1.34 - 41.35	0.96 - 5.15
a_3 - b_3	1.50 - 10.00	0.62 - 3.57	1.09 - 25.68

Como puede apreciarse, únicamente en una de las seis funciones, en f_{Dec}, el sistema ha encontrado como mejor AG a un modelo generacional mientras que en el resto ha hallado modelos estacionarios (steady-state). A su vez, en la mayoría de los casos se trata de un AG Adaptativo y solamente en la función f_{RR} se puede considerar que las probabilidades son casi estáticas y sus valores iniciales se mantienen practicamente constantes.

Las figuras 6.6, 6.7, 6.8, 6.9, 6.10 y 6.11 muestran como varían las probabilidades de selección, cruce y mutación durante la ejecución de 100000 evaluaciones en el AG para la optimización de cada una de las funciones. Puede observarse que la P_s prácticamente mantiene su valor inicial en la mayoría de las funciones y no sufre ningún tipo de atenuación, excepto en la función f_{Sph} donde se atenúa un 40 %. Con respecto a la P_c, se puede decir que tiene unas leves atenuaciones, excepto en f_{Ros} y f_{RR} donde la disminución con respecto a su valor inicial es del 28 % y del 19.5 % respectivamente. Por último, la P_m es el parámetro donde se da la mayor atenuación, ya que se atenúa el 100 % en las funciones f_{Sph} y f_{Ras}, el 28 % en f_{Dec}, el 10 % en f_{One} y el 4 % en f_{Ros} y f_{RR}. Además, la disminución es mucho más acentuada en la etapa inicial de la ejecución del algoritmo. Ello implica que, en primer lugar, se realiza la exploración de la información, aportando diversidad en la población, y posteriormente, se efectúa la explotación, proporcionando convergencia.

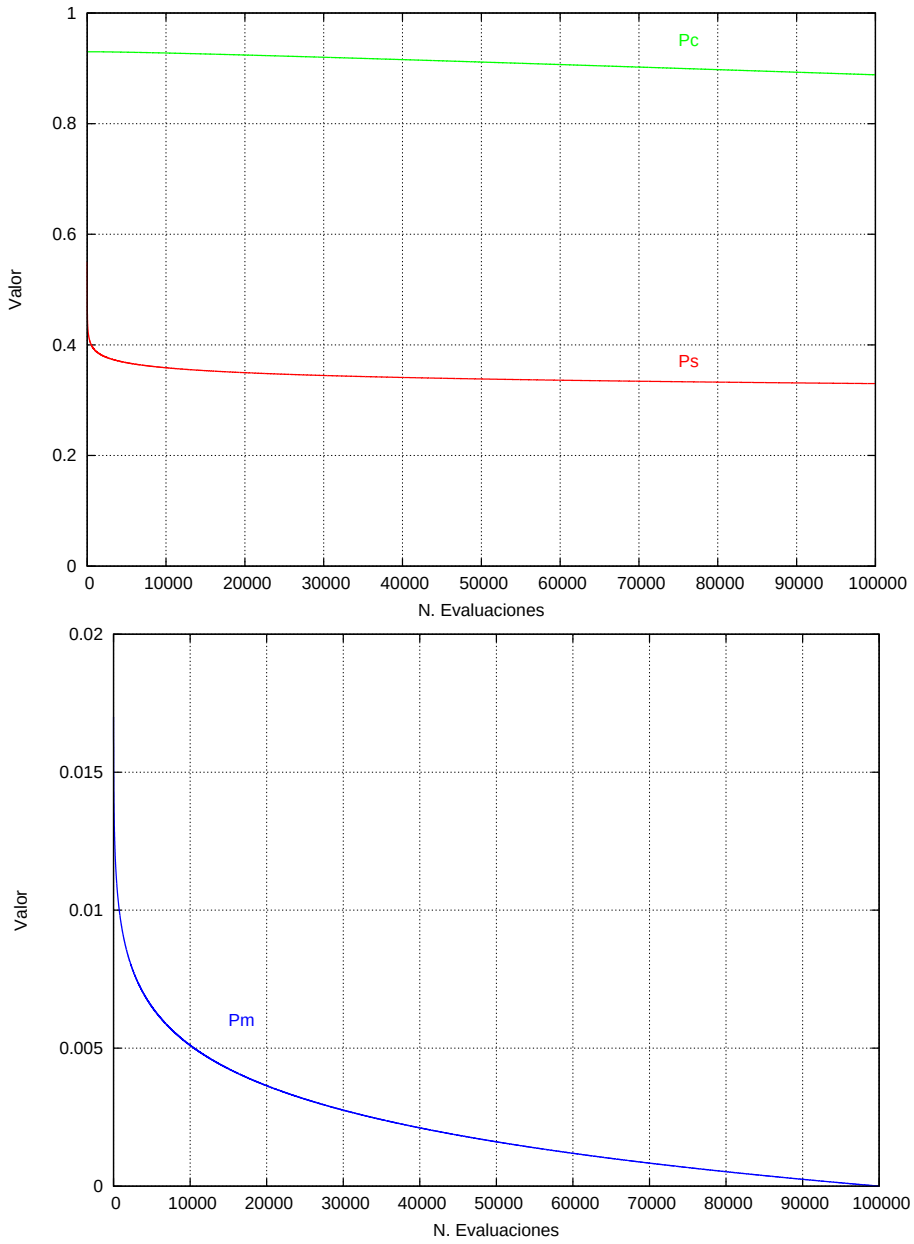

Figura 6.6: *Evolución de P_s, P_c y P_m durante la ejecución del AG en f_{Sph}.*

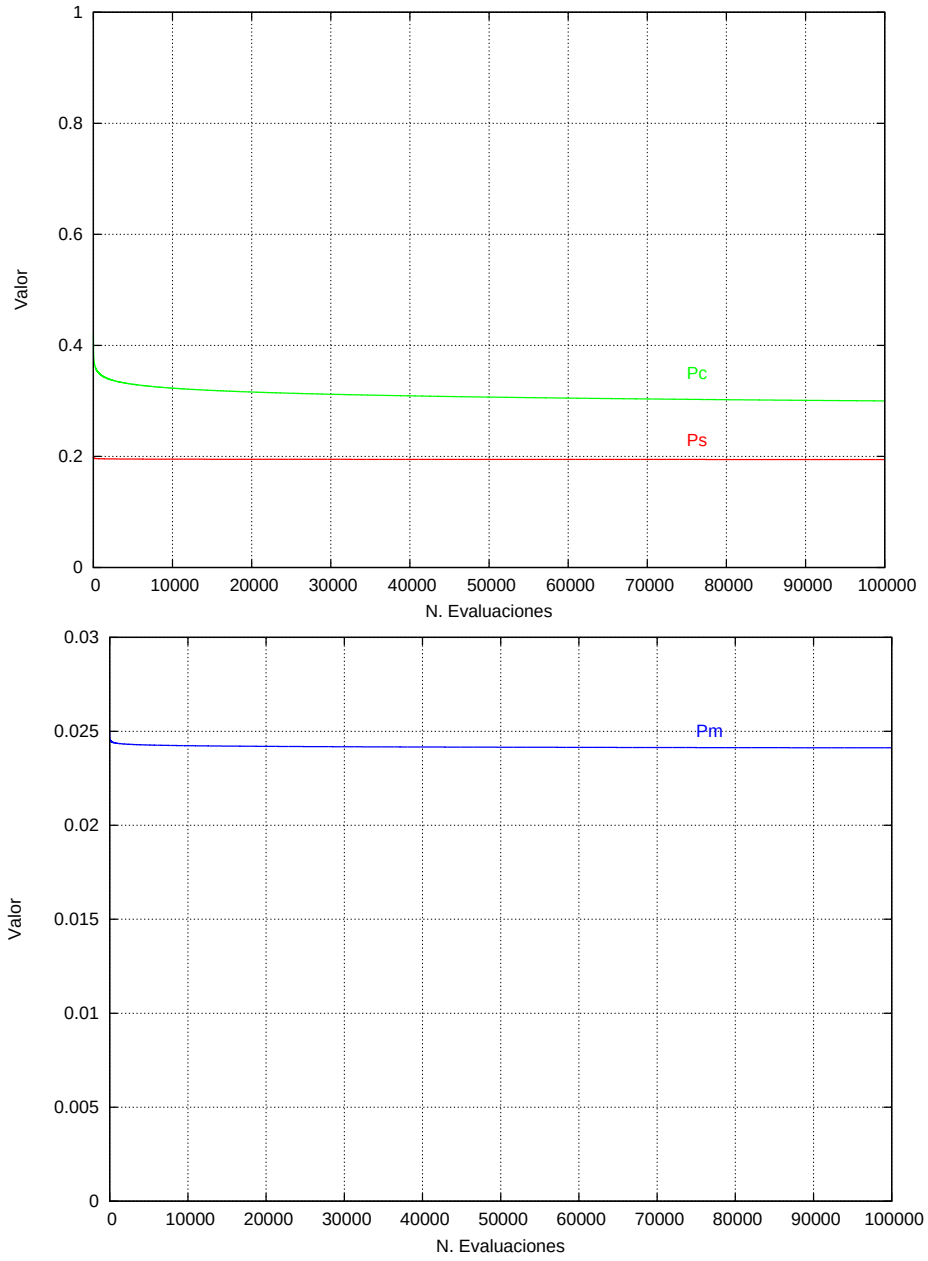

Figura 6.7: *Evolución de P_s, P_c y P_m durante la ejecución del AG en f_{Ros}.*

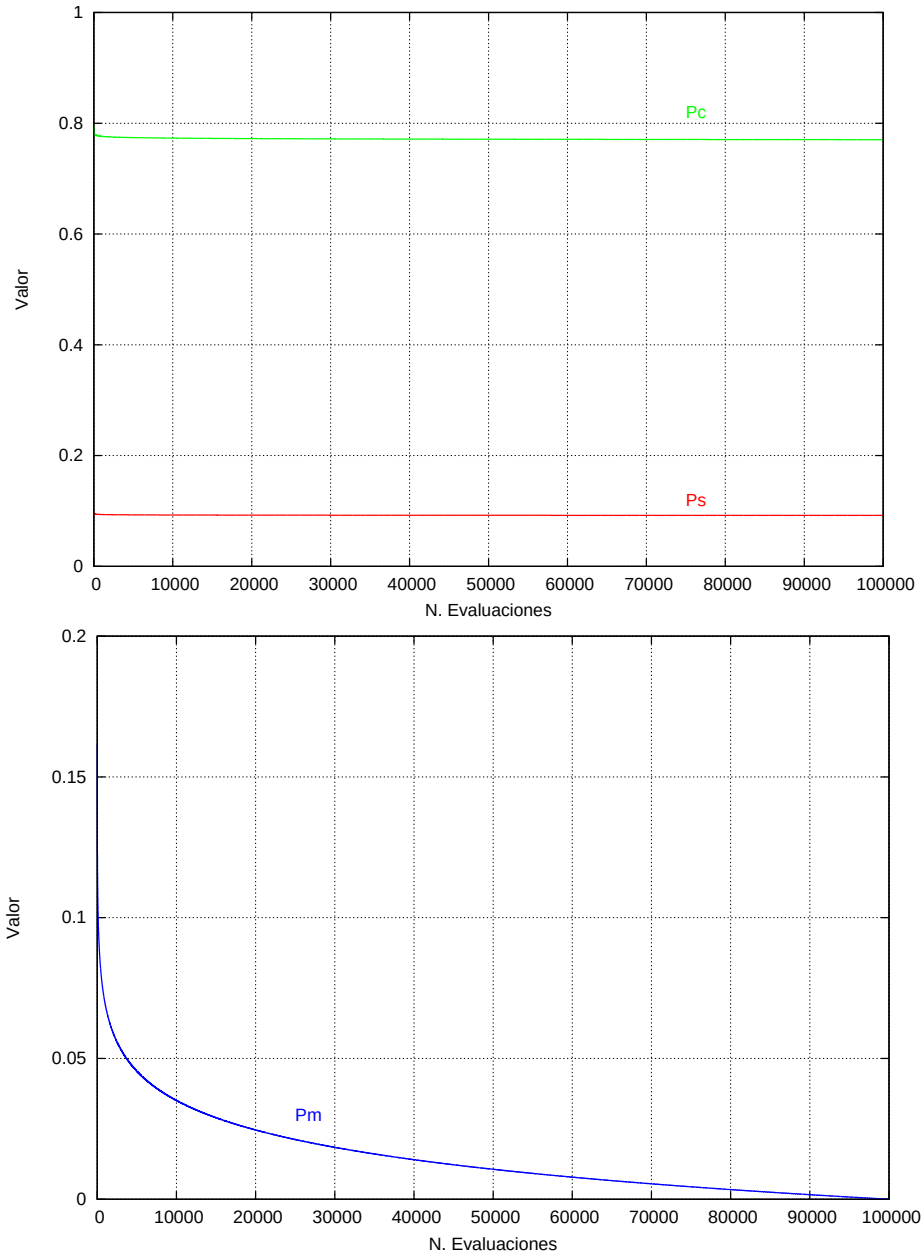

Figura 6.8: *Evolución de P_s, P_c y P_m durante la ejecución del AG en f_{Ras}.*

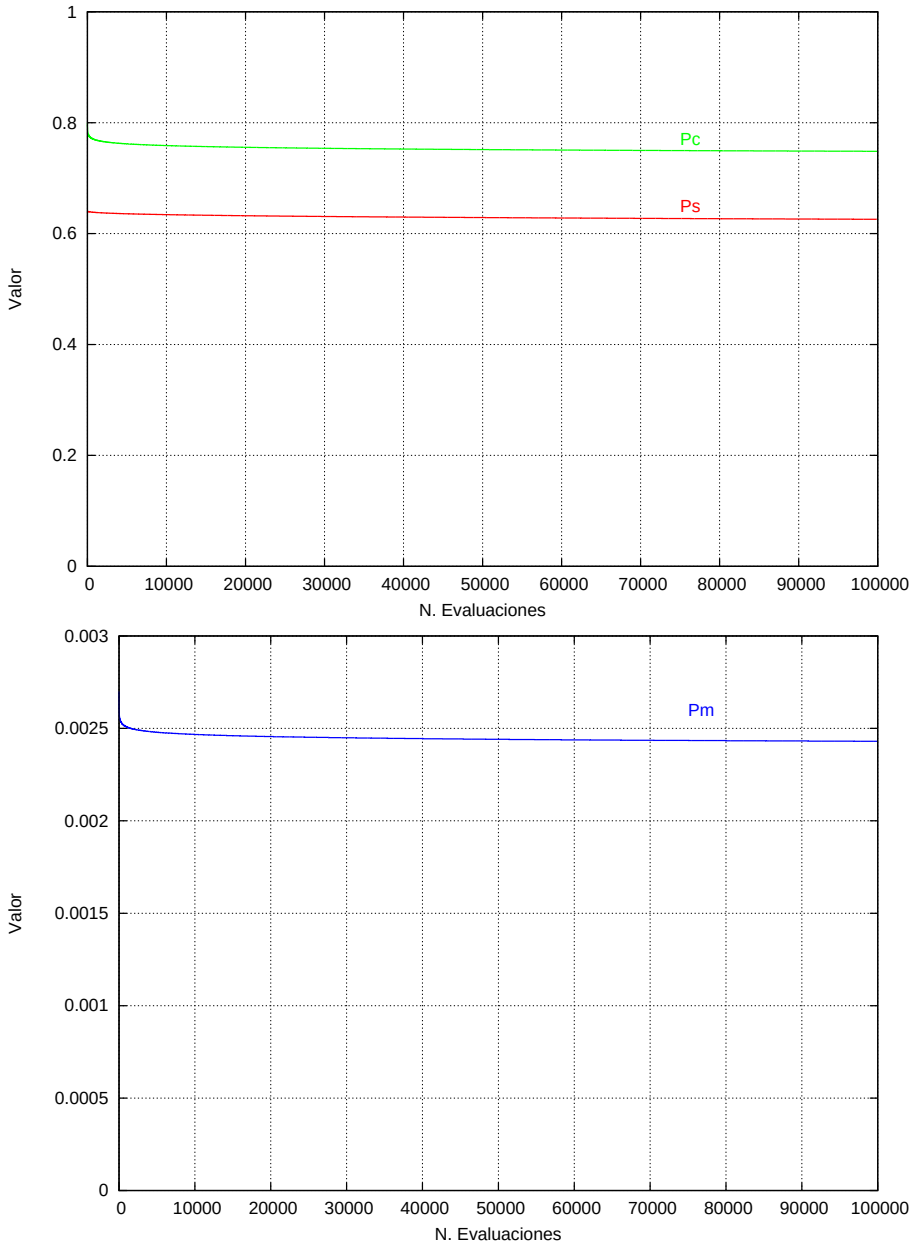

Figura 6.9: *Evolución de P_s, P_c y P_m durante la ejecución del AG en f_{One}.*

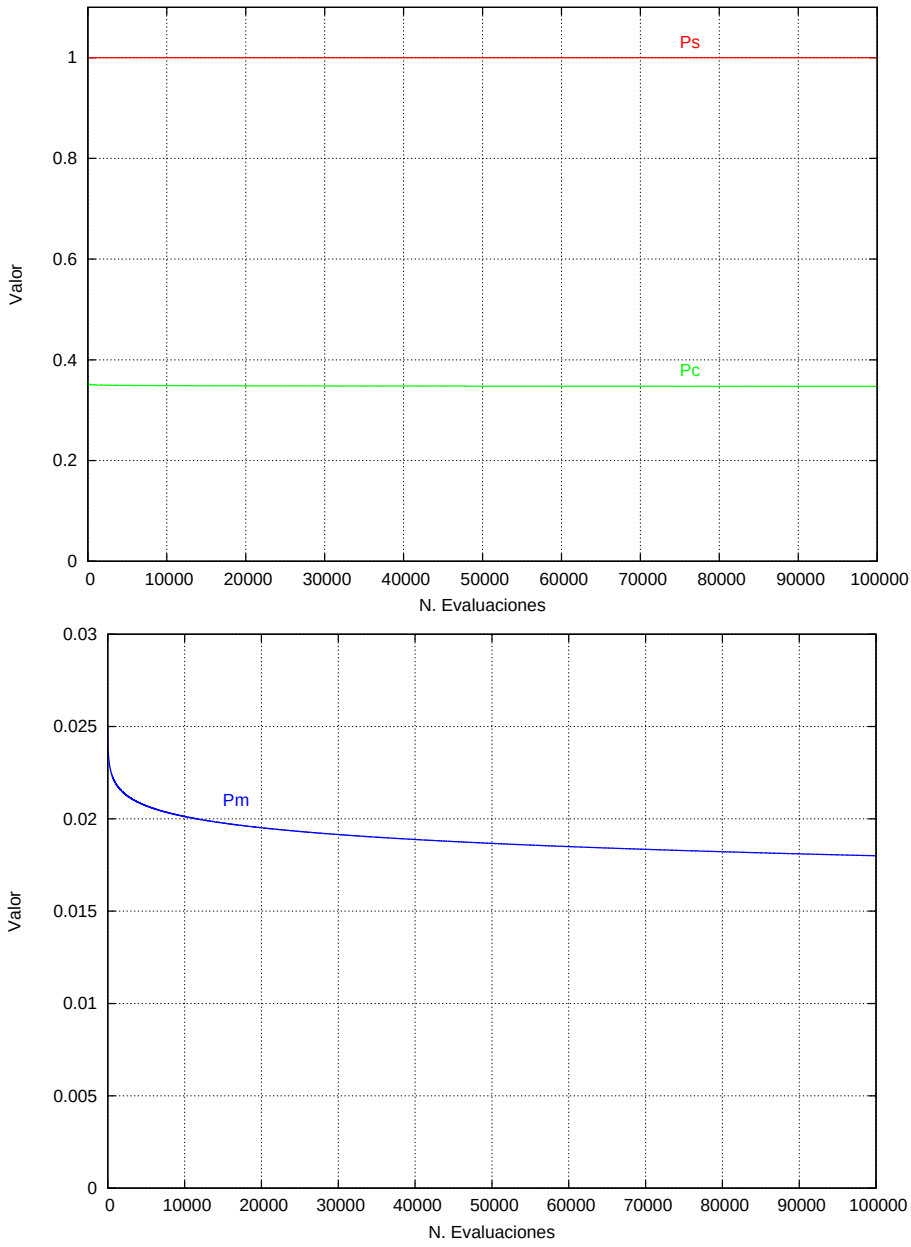

Figura 6.10: *Evolución de P_s, P_c y P_m durante la ejecución del AG en f_{Dec}.*

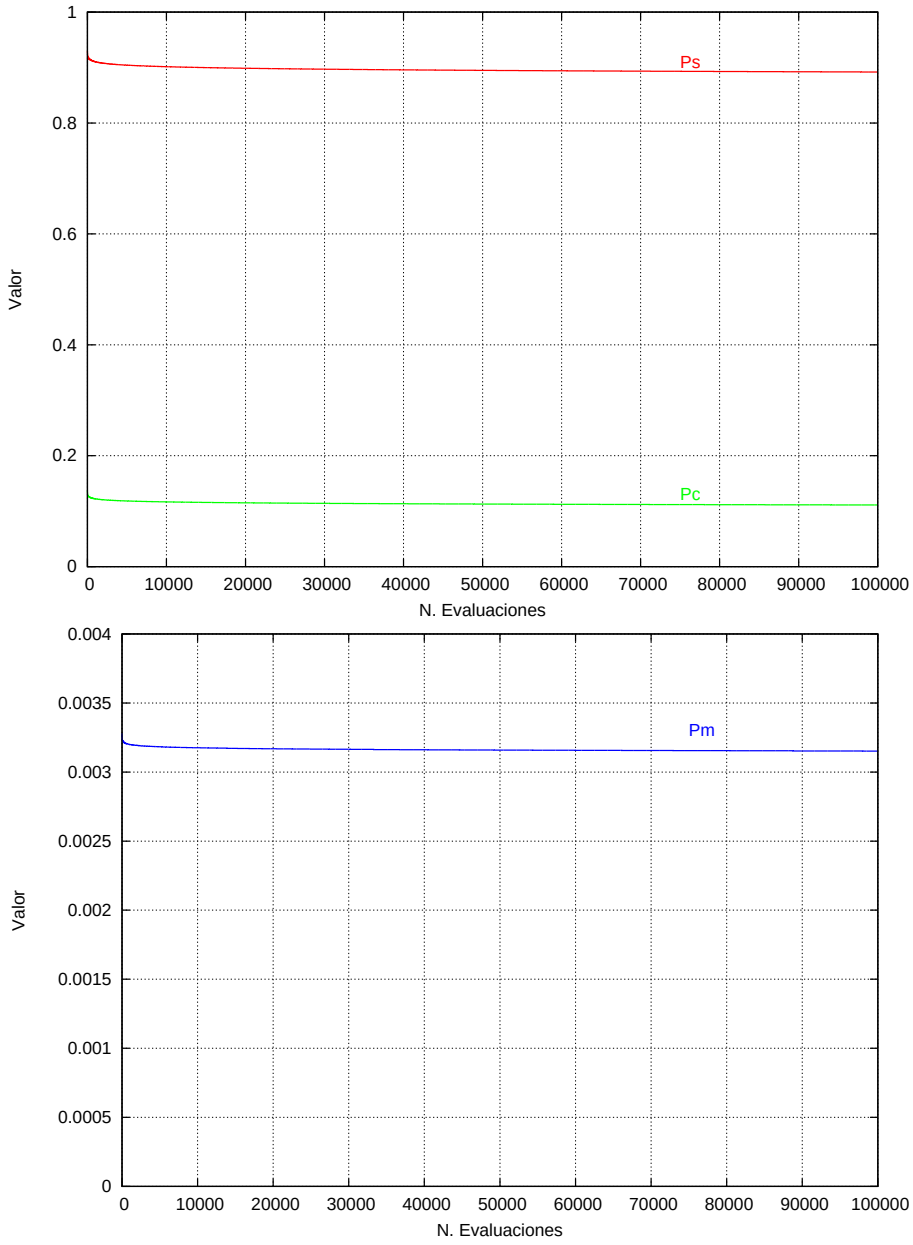

Figura 6.11: *Evolución de P_s, P_c y P_m durante la ejecución del AG en f_{RR}.*

6.5.2. Comportamiento del AG con los parámetros hallados

Una vez hallados los valores de los parámetros para cada una de las funciones se ha comprobado el comportamiento del AG_SGA. Para ello, se ha ejecutado el algoritmo 30 veces con un máximo de 100000 evaluaciones. Las tablas 6.5 y 6.6 muestran los resultados obtenidos. La primera columna muestra el número del experimento realizado; la segunda, la evaluación del mejor individuo obtenido al final de cada ejecución; y la tercera, el número de evaluación a partir de la cual no ha existido una mejora en la solución encontrada. Dado que se trata de un problema de minimización, cuanto menor es la evaluación de los individuos obtenidos mejor es su comportamiento.

A partir de los resultados mostrados podemos indicar, a priori, que el comportamiento del AG_SGA es el siguiente:

- Excelente en las funciones f_{Sph}, f_{Ras} y f_{One} ya que, en todas las ejecuciones, el algoritmo ha encontrado, en cada caso, al mejor individuo posible (cuya evaluación es 2.4E-10, 4.8E-08 y 0.0E+00 respectivamente). Además, el algoritmo ha sido capaz de hallar al mejor individuo con mucha rapidez, siendo necesarias (de media) 12189, 38886 y 10272 evaluaciones respectivamente.

- Muy bueno en la función f_{Dec}, ya que el algoritmo ha hallado en 28 ocasiones el mejor individuo (0.0E+00) y en las otras 2 la evaluación ha sido de 2.0E+00.

- Bueno en la función f_{Ros}, puesto que el algoritmo ha encontrado en 10 ejecuciones al mejor individuo posible (1.5E-09). En las 20 restantes se ha quedado muy cerca de hallarlo pues la evaluación media de los mejores individuos obtenidos ha sido de 2.0E-09.

- Malo en la función f_{RR} ya que en ninguna ocasión el algoritmo ha hallado al mejor individuo (0.0E+00). Además, la media de las evaluaciones de los mejores individuos obtenidos al final de cada ejecución del algoritmo ha sido de 2.4E+01, lo cual indica que AG_SGA ha quedado lejos de encontrar al mejor individuo.

Sin embargo, estas apreciaciones son subjetivas ya que para conocer realmente cómo de bueno o malo es el comportamiento del AG_SGA es necesario compararlo con otros AGs.

Tabla 6.5: *Evaluación de los mejores individuos en las funciones f_{Sph}, f_{Ros} y f_{Ras}.*

	f_{Sph}		f_{Ros}		f_{Ras}	
Nº Exp.	Evaluación	Nº Eval.	Evaluación	Nº Eval.	Evaluación	Nº Eval.
1	2.4E-10	11531	1.5E-09	21472	4.8E-08	40121
2	2.4E-10	12581	3.5E-09	36689	4.8E-08	35504
3	2.4E-10	12581	2.7E-09	51514	4.8E-08	34408
4	2.4E-10	13286	1.7E-09	99617	4.8E-08	34401
5	2.4E-10	10708	1.5E-09	75023	4.8E-08	34803
6	2.4E-10	12567	2.1E-09	53014	4.8E-08	39335
7	2.4E-10	11449	1.7E-09	24628	4.8E-08	31209
8	2.4E-10	12522	1.5E-09	61648	4.8E-08	40081
9	2.4E-10	12426	1.5E-09	17218	4.8E-08	40108
10	2.4E-10	11953	2.7E-09	79920	4.8E-08	38914
11	2.4E-10	11110	1.8E-09	61132	4.8E-08	40108
12	2.4E-10	12660	2.7E-09	41235	4.8E-08	47650
13	2.4E-10	13399	1.7E-09	41806	4.8E-08	47968
14	2.4E-10	12938	1.5E-09	99676	4.8E-08	38945
15	2.4E-10	10997	2.1E-09	20487	4.8E-08	40094
16	2.4E-10	13500	1.8E-09	20278	4.8E-08	35481
17	2.4E-10	11363	1.5E-09	80411	4.8E-08	36647
18	2.4E-10	11344	3.7E-09	99833	4.8E-08	35708
19	2.4E-10	12307	1.7E-09	65896	4.8E-08	48811
20	2.4E-10	12430	1.7E-09	38913	4.8E-08	46205
21	2.4E-10	13365	2.1E-09	84827	4.8E-08	40097
22	2.4E-10	11793	1.7E-09	55329	4.8E-08	34308
23	2.4E-10	11743	2.1E-09	61298	4.8E-08	38924
24	2.4E-10	11782	1.5E-09	23003	4.8E-08	34884
25	2.4E-10	14059	1.7E-09	22574	4.8E-08	47970
26	2.4E-10	12856	2.7E-09	97125	4.8E-08	34948
27	2.4E-10	12816	1.5E-09	97610	4.8E-08	30573
28	2.4E-10	10836	1.5E-09	70043	4.8E-08	32175
29	2.4E-10	11420	1.5E-09	63128	4.8E-08	38917
30	2.4E-10	11351	1.7E-09	82616	4.8E-08	47270
MEDIA	**2.4E-10**	**12189**	**2.0E-09**	**58265**	**4.8E-08**	**38886**

Tabla 6.6: *Evaluación de los mejores individuos en las funciones f_{One}, f_{Dec} y f_{RR}.*

	f_{One}		f_{Dec}		f_{RR}	
Nº Exp.	Evaluación	Nº Eval.	Evaluación	Nº Eval.	Evaluación	Nº Eval.
1	0.0E+00	9654	0.0E+00	99542	2.4E+01	95891
2	0.0E+00	10162	0.0E+00	42193	3.2E+01	92945
3	0.0E+00	8178	0.0E+00	71630	8.0E+00	96517
4	0.0E+00	9882	0.0E+00	84999	1.6E+01	90161
5	0.0E+00	13915	0.0E+00	34984	2.4E+01	79067
6	0.0E+00	9995	0.0E+00	47151	1.6E+01	94256
7	0.0E+00	11345	0.0E+00	24989	3.2E+01	87306
8	0.0E+00	10073	0.0E+00	69965	8.0E+00	91769
9	0.0E+00	10101	2.0E+00	82794	2.4E+01	91114
10	0.0E+00	9136	0.0E+00	62360	2.4E+01	61913
11	0.0E+00	12117	0.0E+00	74528	2.4E+01	75059
12	0.0E+00	11092	0.0E+00	41957	4.0E+01	86640
13	0.0E+00	12556	0.0E+00	49830	1.6E+01	97783
14	0.0E+00	10144	0.0E+00	61436	1.6E+01	74651
15	0.0E+00	8908	0.0E+00	91575	3.2E+01	89805
16	0.0E+00	9942	0.0E+00	88058	4.0E+01	89505
17	0.0E+00	11294	0.0E+00	63147	4.0E+01	90915
18	0.0E+00	10021	0.0E+00	52345	1.6E+01	99085
19	0.0E+00	10273	0.0E+00	50264	8.0E+00	95531
20	0.0E+00	8877	0.0E+00	45897	1.6E+01	96649
21	0.0E+00	9358	0.0E+00	61283	4.0E+01	81744
22	0.0E+00	9118	0.0E+00	41200	3.2E+01	98440
23	0.0E+00	9494	0.0E+00	68199	2.4E+01	83326
24	0.0E+00	9831	0.0E+00	52031	1.6E+01	94705
25	0.0E+00	9309	0.0E+00	67965	3.2E+01	77567
26	0.0E+00	9946	0.0E+00	57205	8.0E+00	75854
27	0.0E+00	11910	0.0E+00	61344	1.6E+01	96685
28	0.0E+00	10530	2.0E+00	34085	4.0E+01	94437
29	0.0E+00	11762	0.0E+00	52482	2.4E+01	93347
30	0.0E+00	9366	0.0E+00	42721	2.4E+01	98885
MEDIA	**0.0E+00**	**10272**	**1.3E-01**	**59272**	**2.4E+01**	**89052**

6.5.3. Comportamiento de otros AGs Adaptativos

En este apartado se muestran los resultados obtenidos al utilizar los principales métodos de adaptación sobre la probabilidad de mutación. Estos métodos han sido adaptados para que la P_m varíe en el intervalo [0.001, 0.01], que abarca el rango de valores más frecuentemente utilizados en la literatura. Todos los algoritmos se han ejecutado 30 veces con un máximo de 100000 evaluaciones. Los mecanismos adaptativos utilizados han sido los siguientes:

- Control determinístico de P_m (AG_DET), donde la probabilidad de mutación ha sido modificada atendiendo a una regla determinística, la cual ha modificado su valor sin utilizar ningún tipo de realimentación del estado de la búsqueda. La función lineal para controlar el decremento de P_m ha sido la indicada en [Herrera y Lozano, 2003], por lo que P_m ha variado de la siguiente forma:

$$P_m(t) = \ P_m^h - \frac{P_m^h - P_m^l}{T} t \quad 0 \leq \mathrm{t} \leq \mathrm{T}$$

  donde P_m^h=0.01, P_m^l=0.001 y T=100000.

- Control adaptativo a nivel del individuo de P_m (AG_IL), donde la probabilidad de mutación ha sido modificada en función de la evaluación de cada individuo y de la evaluación del resto de individuos de la población. Cada individuo ha tenido asociada una P_m que ha variado según la siguiente fórmula:

$$P_m = \begin{cases} 0.01 & \text{si } f' > \overline{f} \\ 0.01 * \frac{f' - f_{min}}{\overline{f} - f_{min}} & \text{si } f_{min} < f' \leq \overline{f} \\ 0.001 & \text{si } f' = f_{min} \end{cases}$$

  donde f' es la evaluación del individuo, f_{min} es la evaluación del mejor individuo en la población y $\overline{f}$ es la evaluación media de la población.

- Control adaptativo a nivel de la población de P_m, en el que se ha utilizado un controlador borroso para adaptar la probabilidad de mutación durante la ejecución del AG. Este sistema es el denominado *Fuzzy Adaptive Mutation Probability* (AG_FAMP). El controlador borroso se ha disparado cada G=50 iteraciones y la P_m ha permanecido fija durante ese tiempo.

- Control auto-adaptativo a nivel del individuo de P_m (AG_SELF), donde la probabilidad de mutación ha sido codificada junto a los individuos de la población y ha evolucionado junto a ellos. Para la configuración del algoritmo se han elegido los valores de δ=0.001, P_m^l =0.001 y P_m^h=0.01.

- Control aleatorio (AG_RAN), donde en cada iteración las probabilidades de mutación han sido elegidas aleatoriamente en el intervalo [0.001, 0.01].

También se ha considerado interesante comprobar el comportamiento del AG utilizando probabilidades de mutación estáticas durante su ejecución. Las probabilidades utilizadas han sido las siguientes:

1. P_m=0.001 (AG_EST1)
2. P_m=0.005 (AG_EST2)
3. P_m=0.01 (AG_EST3)

En todos los casos, las probabilidades de selección P_s y de cruce P_c han sido las recomendadas en [DeJong, 1975]: P_s=1.0 y P_c=0.6.

La tabla 6.7 muestra un resumen de los resultados obtenidos. La primera columna indica el algoritmo utilizado. Posteriormente, para cada función, existen tres columnas identificadas como ME, MNE y NO. La columna ME muestra la media de las evaluaciones de los mejores individuos obtenidos al final de cada ejecución del algoritmo, la columna MNE muestra la media del número de evaluaciones a partir de la cual no ha existido una mejora en la solución encontrada y, por último, la columna NO muestra el número de veces que el AG ha encontrado al mejor individuo posible.

Tabla 6.7: *Resumen de los resultados obtenidos en las funciones de prueba.*

	f_{Sph}			f_{Ros}			f_{Ras}		
Algoritmo	ME	MNE	NO	ME	MNE	NO	ME	MNE	NO
AG_EST1	2.4E-10	34350	30	1.2E-01	99885	0	7.6E+00	74544	0
AG_EST2	9.7E-08	98139	0	2.8E-03	88366	4	8.9E-05	98511	0
AG_EST3	9.0E-06	98326	0	2.1E-07	50088	29	7.9E-02	99336	0
AG_RAN	1.8E-07	97553	0	9.6E-04	84255	13	1.3E-04	98600	0
AG_DET	3.9E-10	97508	21	4.7E-04	56256	10	3.3E-02	98435	21
AG_IL	2.4E-10	61502	30	1.0E-01	99772	0	2.6E+00	93311	2
AG_SELF	2.7E-10	85631	26	2.3E-02	96908	0	4.0E-01	93782	1
AG_FAMP	2.4E-10	40255	30	1.9E-04	76056	19	5.0E-08	70484	28
	f_{One}			f_{Dec}			f_{RR}		
Algoritmo	ME	MNE	NO	ME	MNE	NO	ME	MNE	NO
AG_EST1	0.0E+00	21685	30	9.9E+00	1926	0	4.3E+01	87511	0
AG_EST2	9.8E+00	93101	0	5.5E+00	54596	0	2.8E+01	78840	0
AG_EST3	3.3E+01	94420	0	0.0E+00	22296	30	7.1E+01	69101	0
AG_RAN	1.2E+01	94041	0	3.6E+00	64480	5	3.5E+01	69064	0
AG_DET	6.7E-02	95262	28	1.9E+00	36432	12	2.5E+01	90665	1
AG_IL	6.8E+00	94246	0	8.1E+00	32645	0	4.6E+01	85548	0
AG_SELF	4.0E-01	66359	23	6.7E+00	33470	0	2.2E+01	84624	0
AG_FAMP	0.0E+00	29997	30	9.3E-01	64836	22	3.4E+01	82308	0

A continuación, las siguientes tablas muestran, de forma más detallada, los resultados obtenidos por los distintos AGs en cada una de las funciones de prueba. Para cada experimento, se muestra la evaluación del mejor individuo obtenido y el número de evaluación a partir de la cual no ha existido mejora en la solución encontrada.

Tabla 6.8: *Resultados obtenidos en la función Sphere Model (f_{Sph})*

Función Sphere Model (f_{Sph})								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	2.4E-10	32414	1.6E-08	99483	9.5E-06	96355	4.3E-07	96130
2	2.4E-10	34905	2.4E-08	98448	3.5E-05	99171	7.2E-08	99531
3	2.4E-10	33778	2.0E-08	93198	2.8E-06	97292	4.5E-07	99611
4	2.4E-10	31660	1.8E-07	99878	4.4E-06	96739	4.5E-08	99273
5	2.4E-10	35353	2.6E-07	100000	8.8E-06	99180	8.7E-08	98795
6	2.4E-10	34203	3.3E-08	99812	3.3E-06	97134	1.2E-07	89450
7	2.4E-10	30352	5.7E-08	99120	1.3E-05	99474	2.5E-07	98817
8	2.4E-10	33704	2.9E-08	99738	2.5E-06	93992	1.5E-07	91849
9	2.4E-10	34209	7.5E-08	96512	1.1E-05	99532	4.0E-07	99862
10	2.4E-10	32554	3.7E-08	97103	9.6E-06	99903	1.2E-06	99430
11	2.4E-10	35507	3.7E-09	92012	3.1E-06	98529	8.3E-08	98002
12	2.4E-10	30441	4.5E-08	98857	1.0E-05	99821	1.3E-07	98940
13	2.4E-10	29652	4.5E-08	96205	2.2E-06	99538	1.3E-07	99589
14	2.4E-10	36171	3.7E-08	99004	6.3E-06	99752	5.4E-08	99471
15	2.4E-10	33198	5.4E-07	99918	7.0E-06	99198	9.0E-08	99268
16	2.4E-10	29916	3.0E-07	97190	4.5E-06	99470	7.5E-07	95504
17	2.4E-10	37509	3.2E-08	98983	3.1E-06	99093	2.1E-08	98108
18	2.4E-10	37808	4.9E-08	99325	2.5E-05	99945	2.5E-08	92920
19	2.4E-10	39936	2.9E-08	99656	5.3E-06	98302	1.6E-07	99072
20	2.4E-10	35168	9.6E-08	99308	4.2E-06	96926	4.1E-08	99480
21	2.4E-10	36316	1.1E-08	94608	5.2E-05	99446	1.1E-08	99296
22	2.4E-10	34901	1.7E-07	98677	2.2E-06	99430	3.8E-08	96703
23	2.4E-10	40614	1.0E-07	97469	1.2E-06	95202	3.4E-08	99933
24	2.4E-10	35292	4.0E-07	99741	2.1E-06	99839	6.5E-08	98571
25	2.4E-10	33898	3.1E-08	99462	3.2E-07	96855	1.6E-09	89454
26	2.4E-10	34517	3.1E-08	98691	2.5E-05	98290	2.4E-07	98546
27	2.4E-10	37347	1.7E-07	97959	8.8E-06	98727	5.3E-08	94592
28	2.4E-10	32414	2.9E-08	96038	2.9E-06	97452	1.4E-08	99831
29	2.4E-10	31346	5.6E-08	98218	1.9E-06	98441	2.6E-08	97298
30	2.4E-10	35425	2.4E-08	99548	2.7E-06	96747	1.5E-07	99257
MEDIA	**2.4E-10**	**34350**	**9.7E-08**	**98139**	**9.0E-06**	**98326**	**1.8E-07**	**97553**

Tabla 6.9: *Resultados obtenidos en la función Sphere Model (f_{Sph})*

Función Sphere Model (f_{Sph})								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	8.1E-10	98987	2.4E-10	63367	2.4E-10	93741	2.4E-10	31699
2	2.4E-10	94435	2.4E-10	62364	2.4E-10	98053	2.4E-10	31884
3	2.4E-10	96326	2.4E-10	61689	2.4E-10	70451	2.4E-10	34198
4	2.4E-10	97980	2.4E-10	60021	2.4E-10	85571	2.4E-10	37923
5	2.4E-10	99715	2.4E-10	60770	2.4E-10	80838	2.4E-10	30813
6	2.4E-10	94678	2.4E-10	54055	2.4E-10	72188	2.4E-10	34350
7	4.3E-10	99946	2.4E-10	68079	4.3E-10	94465	2.4E-10	30594
8	2.4E-10	97555	2.4E-10	72699	2.4E-10	77523	2.4E-10	34646
9	1.0E-09	97532	2.4E-10	74574	2.4E-10	70279	2.4E-10	37657
10	2.4E-10	98182	2.4E-10	59486	2.4E-10	98854	2.4E-10	92549
11	2.4E-10	96786	2.4E-10	55527	4.3E-10	96046	2.4E-10	46150
12	1.6E-09	99973	2.4E-10	54648	6.2E-10	99069	2.4E-10	34495
13	2.4E-10	98182	2.4E-10	54418	2.4E-10	99805	2.4E-10	34014
14	2.4E-10	99406	2.4E-10	66326	4.3E-10	87497	2.4E-10	33174
15	2.4E-10	93044	2.4E-10	66836	2.4E-10	85923	2.4E-10	35830
16	2.4E-10	97350	2.4E-10	56144	2.4E-10	83744	2.4E-10	32930
17	2.4E-10	94630	2.4E-10	65918	2.4E-10	70452	2.4E-10	37066
18	2.4E-10	97702	2.4E-10	45810	2.4E-10	79966	2.4E-10	46915
19	4.3E-10	99775	2.4E-10	49701	2.4E-10	70451	2.4E-10	55234
20	2.4E-10	98412	2.4E-10	70399	2.4E-10	92393	2.4E-10	34094
21	6.2E-10	99936	2.4E-10	64309	2.4E-10	92647	2.4E-10	40089
22	2.4E-10	95947	2.4E-10	69685	2.4E-10	92913	2.4E-10	42154
23	2.4E-10	94332	2.4E-10	61709	2.4E-10	68515	2.4E-10	75382
24	4.3E-10	98961	2.4E-10	62695	2.4E-10	79674	2.4E-10	30606
25	4.3E-10	98935	2.4E-10	56447	2.4E-10	93661	2.4E-10	34766
26	2.4E-10	97609	2.4E-10	60460	2.4E-10	88073	2.4E-10	57277
27	2.4E-10	97449	2.4E-10	64185	2.4E-10	79654	2.4E-10	39908
28	2.4E-10	94960	2.4E-10	63367	2.4E-10	70862	2.4E-10	32284
29	8.1E-10	99807	2.4E-10	52286	2.4E-10	97678	2.4E-10	34331
30	2.4E-10	96714	2.4E-10	67095	2.4E-10	97930	2.4E-10	34650
MEDIA	**3.9E-10**	**97508**	**2.4E-10**	**61502**	**2.7E-10**	**85631**	**2.4E-10**	**40255**

Tabla 6.10: *Resultados obtenidos en la función Rosenbrock (f_{Ros})*

Función Rosenbrock (f_{Ros})								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	4.5E-02	99761	8.3E-03	99302	1.5E-09	59227	1.5E-09	99932
2	1.7E-02	99958	8.6E-09	68943	1.5E-09	43442	1.5E-09	70667
3	3.1E-02	100000	1.3E-03	99733	1.5E-09	55333	2.1E-09	89849
4	2.5E-01	99660	3.9E-03	99678	1.5E-09	24806	1.5E-09	98706
5	1.7E-02	99988	2.0E-04	100000	1.5E-09	37564	1.3E-02	99720
6	2.7E-01	99899	4.0E-06	99331	1.5E-09	30455	2.1E-09	69221
7	1.8E-02	99923	1.5E-09	60393	1.5E-09	37466	1.5E-09	29030
8	1.4E-01	99878	1.1E-02	99907	6.3E-06	99835	7.1E-09	97155
9	3.0E-03	99977	1.2E-05	99794	1.5E-09	67680	1.5E-09	18849
10	1.3E-02	99971	3.5E-03	99991	1.5E-09	63542	1.5E-09	95260
11	1.3E-03	99915	1.2E-02	99954	1.5E-09	52394	1.5E-09	75950
12	3.8E-02	99831	1.1E-04	99969	1.5E-09	39833	8.3E-09	94419
13	3.9E-03	99959	4.8E-03	99864	1.5E-09	81137	8.3E-09	74863
14	2.4E-01	99644	5.5E-03	99554	1.5E-09	55689	1.6E-09	97945
15	3.0E-01	99369	2.7E-09	76804	1.5E-09	52899	2.7E-09	65940
16	1.3E-02	99871	2.7E-09	95244	1.5E-09	15928	1.5E-09	84912
17	4.1E-03	99940	1.5E-09	58178	1.5E-09	21455	4.6E-06	99858
18	1.2E-01	99997	2.7E-09	96410	1.5E-09	41601	5.6E-03	100000
19	9.2E-01	99994	1.5E-09	67295	1.5E-09	46283	4.2E-03	99773
20	7.5E-02	99932	2.5E-05	99949	1.5E-09	42307	4.1E-03	100000
21	5.9E-04	99905	4.1E-08	98665	1.5E-09	42350	1.5E-09	96625
22	1.6E-01	99956	1.7E-09	44078	1.5E-09	45941	1.5E-09	85993
23	1.1E-01	99773	1.5E-09	26338	1.5E-09	46551	1.5E-09	95243
24	1.6E-01	100000	7.3E-09	97402	1.5E-09	48076	1.5E-03	99724
25	4.9E-03	100000	5.5E-09	95932	1.5E-09	40426	1.7E-09	71721
26	3.0E-02	99797	8.6E-09	68943	1.5E-09	50507	1.5E-09	49498
27	8.0E-03	99999	5.0E-03	99910	1.5E-09	74908	1.5E-09	79799
28	5.1E-01	99952	5.3E-04	99683	1.5E-09	71323	2.7E-09	99033
29	9.7E-02	99788	1.9E-03	99960	1.5E-09	79530	1.7E-09	88069
30	1.4E-02	99901	2.5E-02	99775	1.5E-09	34165	4.4E-04	99888
MEDIA	**1.2E-01**	**99885**	**2.8E-03**	**88366**	**2.1E-07**	**50088**	**9.6E-04**	**84255**

Tabla 6.11: *Resultados obtenidos en la función Rosenbrock (f_{Ros})*

Función Rosenbrock (f_{Ros})								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	2.7E-05	99920	4.7E-02	99333	4.1E-02	100000	1.5E-09	95918
2	1.7E-09	70019	5.4E-02	99340	2.7E-02	100000	2.7E-09	97714
3	1.6E-09	53747	2.4E-01	99818	1.6E-02	99914	7.8E-08	99145
4	6.0E-03	99795	1.5E-01	99855	6.8E-02	99884	1.5E-09	61423
5	2.7E-03	99956	1.2E-01	99716	3.5E-06	99964	1.0E-07	95946
6	1.5E-09	30877	1.5E-01	99776	9.4E-05	99849	1.5E-09	39111
7	4.4E-04	99691	2.5E-01	99621	8.9E-03	99935	1.5E-09	79204
8	3.5E-04	99980	8.3E-03	99763	6.7E-03	99913	1.5E-09	72746
9	1.7E-09	56073	3.2E-01	99859	9.4E-02	100000	1.5E-09	45893
10	1.7E-09	44127	4.2E-04	99747	1.2E-02	100000	3.7E-03	99965
11	2.7E-09	64435	5.1E-02	99825	5.7E-03	99938	2.5E-04	99954
12	1.5E-09	29069	1.3E-02	99851	8.4E-03	99924	1.5E-09	37041
13	1.8E-09	37413	2.5E-02	99786	7.2E-03	99913	1.5E-09	62542
14	1.5E-09	12534	4.7E-03	99804	6.0E-03	99999	1.8E-03	99879
15	1.5E-09	39885	4.6E-02	99434	7.0E-03	99771	1.5E-09	37000
16	3.8E-04	99842	2.5E-03	99560	7.5E-03	99850	1.5E-09	68205
17	1.5E-09	66713	8.8E-05	99998	5.9E-03	99984	1.7E-09	95407
18	1.5E-09	50462	2.1E-02	99756	6.2E-07	84453	8.3E-09	95595
19	1.5E-09	44233	2.0E-01	99909	1.5E-01	99834	1.5E-09	96536
20	2.1E-09	38569	6.3E-02	99933	2.5E-02	99974	1.5E-09	80897
21	3.4E-09	49325	1.7E-02	99931	1.0E-02	99921	1.5E-09	46641
22	1.5E-09	15722	4.5E-01	99973	1.3E-01	99675	1.1E-06	96742
23	8.3E-09	50626	1.8E-02	100000	8.3E-09	85207	1.5E-09	82608
24	1.5E-09	40224	1.6E-02	99993	3.4E-09	40217	1.5E-09	50628
25	3.7E-09	59460	1.2E-02	99444	9.5E-03	99836	1.7E-09	97624
26	1.6E-09	34536	7.0E-03	100000	2.1E-07	99676	1.5E-09	64318
27	1.6E-09	35099	4.4E-04	99795	1.4E-02	100000	1.5E-09	61484
28	1.6E-09	39671	1.2E-02	99875	7.1E-03	99981	1.5E-09	64590
29	1.5E-09	25823	7.8E-01	99873	1.5E-02	99723	1.7E-09	95407
30	4.0E-03	99858	2.3E-05	99586	5.9E-08	99919	1.5E-09	61523
MEDIA	**4.7E-04**	**56256**	**1.0E-01**	**99772**	**2.3E-02**	**96908**	**1.9E-04**	**76056**

Tabla 6.12: *Resultados obtenidos en la función Rastrigin (f_{Ras})*

Función Rastrigin (f_{Ras})								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	5.0E+00	99237	1.1E-05	98176	5.4E-03	98974	9.2E-05	99708
2	9.0E+00	99639	2.6E-05	99624	1.9E-03	99711	9.1E-06	99532
3	7.0E+00	62136	3.2E-04	99231	1.3E-03	99916	6.6E-06	97587
4	6.0E+00	59454	8.1E-06	97183	6.7E-03	99765	3.6E-04	99354
5	9.0E+00	24899	5.4E-05	99521	1.2E-03	99752	6.5E-06	99379
6	5.0E+00	78739	5.8E-05	98123	5.4E-04	98940	2.4E-04	98569
7	9.9E+00	75744	1.5E-05	99383	4.3E-04	99996	3.1E-05	94268
8	4.0E+00	90008	3.5E-05	98587	1.6E-03	99562	4.0E-06	99784
9	5.0E+00	90220	2.7E-05	97874	3.5E-04	96635	1.4E-04	100000
10	6.0E+00	77552	7.6E-05	99634	2.6E-02	99362	2.3E-05	99904
11	6.0E+00	82338	6.8E-06	96221	2.1E-03	99990	3.4E-04	99144
12	1.9E+01	86620	1.4E-06	93362	2.1E-04	99823	5.0E-05	97112
13	8.0E+00	92322	4.2E-06	97240	2.2E-03	99740	7.1E-05	99147
14	6.0E+00	79041	1.8E-04	99402	2.4E-03	98767	1.4E-05	98794
15	6.0E+00	62171	1.9E-06	98112	5.9E-04	99915	2.6E-04	99346
16	9.9E+00	55747	3.8E-04	99909	1.7E-03	97536	6.0E-05	99949
17	8.0E+00	57365	2.7E-04	98733	5.5E-04	99159	9.0E-05	98792
18	7.0E+00	86461	2.4E-04	100000	3.7E-03	99715	6.6E-06	98908
19	4.1E+00	99850	4.7E-04	99747	2.4E-03	97725	7.4E-06	99254
20	3.0E+00	81804	1.2E-04	99956	1.3E-03	99470	1.8E-04	99584
21	9.9E+00	88686	9.8E-06	96688	8.6E-04	99362	8.9E-05	99685
22	7.0E+00	64700	3.5E-06	99352	3.2E-03	99946	2.3E-05	96290
23	1.5E+01	40319	1.9E-04	98879	1.0E+00	99518	3.2E-05	98888
24	2.0E+00	85362	2.1E-05	98913	5.3E-03	99034	5.0E-05	97845
25	6.0E+00	95821	8.5E-06	98417	2.4E-02	99803	5.9E-05	98893
26	9.9E+00	39837	7.1E-05	97837	1.0E+00	99213	2.4E-05	97670
27	1.1E+01	28960	2.9E-05	99414	4.0E-04	99608	1.6E-03	99568
28	9.0E+00	89791	3.8E-05	99738	9.4E-04	99547	2.0E-06	91580
29	3.0E+00	89904	2.0E-06	97863	3.3E-03	99653	3.0E-05	99482
30	1.4E+01	71597	2.1E-06	98212	2.7E-01	99948	1.3E-04	99970
MEDIA	**7.6E+00**	**74544**	**8.9E-05**	**98511**	**7.9E-02**	**99336**	**1.3E-04**	**98600**

Tabla 6.13: *Resultados obtenidos en la función Rastrigin (f_{Ras})*

Función Rastrigin (f_{Ras})								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	4.8E-08	99972	2.0E+00	89873	8.5E-08	89094	4.8E-08	59321
2	1.2E-07	99451	2.4E-04	98781	4.8E-08	88814	4.8E-08	83772
3	4.8E-08	99972	9.9E-01	91347	9.9E-01	86781	4.8E-08	55014
4	4.8E-08	97781	3.0E+00	89297	3.1E-07	93439	4.8E-08	54228
5	4.8E-08	96420	9.9E-01	89696	1.1E-06	99674	4.8E-08	63318
6	4.8E-08	98725	7.0E+00	64520	8.5E-08	94776	4.8E-08	59321
7	1.2E-07	99846	4.2E+00	99265	4.0E+00	100000	4.8E-08	64565
8	4.8E-08	97894	4.8E-08	83255	5.1E-06	99094	4.8E-08	99348
9	4.8E-08	98725	4.0E+00	82254	1.4E-06	98917	4.8E-08	49186
10	4.8E-08	97646	1.1E+00	99929	1.6E-07	95988	4.8E-08	98594
11	8.5E-08	99345	7.0E+00	69891	9.9E-01	58865	4.8E-08	89970
12	4.8E-08	98778	2.0E+00	78731	8.5E-08	89804	4.8E-08	87545
13	8.5E-08	98509	2.0E+00	74666	5.5E-03	98587	8.51E-08	71705
14	4.8E-08	95224	1.0E+00	99852	5.4E-07	98722	8.51E-08	60510
15	4.8E-08	96523	2.1E+00	99849	2.0E-07	97830	4.8E-08	94384
16	4.8E-08	97876	4.0E+00	99518	1.7E-05	99018	4.8E-08	92911
17	9.9E-01	96418	6.0E+00	87024	2.4E-07	98755	4.8E-08	68299
18	4.8E-08	96256	5.0E+00	99364	8.4E-07	99777	4.8E-08	46578
19	1.2E-07	99705	1.0E+00	99482	3.1E-07	94229	4.8E-08	73281
20	4.8E-08	97357	3.0E+00	99668	1.2E-07	100000	4.8E-08	68023
21	4.8E-08	95250	2.0E+00	88167	1.2E-07	97147	4.8E-08	76379
22	4.8E-08	99587	3.0E+00	98254	5.0E+00	59098	4.8E-08	87605
23	2.7E-07	99711	4.8E-08	92156	9.9E-01	98269	4.8E-08	54098
24	4.8E-08	99623	5.0E+00	79533	2.0E-07	97472	4.8E-08	54304
25	4.8E-08	99913	1.0E+00	97623	2.0E-07	95959	4.8E-08	57967
26	4.8E-08	98778	8.0E-07	99501	4.6E-07	99601	4.8E-08	67692
27	4.8E-08	99346	6.0E+00	89953	3.5E-07	99788	4.8E-08	60472
28	8.5E-08	99749	5.0E+00	97991	8.5E-08	85919	4.8E-08	96854
29	1.2E-07	99947	2.4E-01	99928	8.5E-08	98472	4.8E-08	53443
30	4.8E-08	98725	9.9E-01	99958	5.8E-07	99572	4.8E-08	65824
MEDIA	**3.3E-02**	**98435**	**2.6E+00**	**93311**	**4.0E-01**	**93782**	**5.0E-08**	**70484**

Tabla 6.14: *Resultados obtenidos en la función One-Max (f_{One})*

Función One-Max (f_{One})								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	0.0E+00	24399	1.1E+01	92972	3.1E+01	96258	1.2E+01	85147
2	0.0E+00	23483	9.0E+00	91518	3.9E+01	98408	6.0E+00	95352
3	0.0E+00	22535	6.0E+00	96564	3.7E+01	96163	1.3E+01	93380
4	0.0E+00	22750	8.0E+00	99033	3.8E+01	83011	1.2E+01	98895
5	0.0E+00	18517	1.0E+01	98882	3.8E+01	93276	1.3E+01	97582
6	0.0E+00	18992	1.5E+01	97387	3.1E+01	90774	1.2E+01	98330
7	0.0E+00	21588	8.0E+00	90270	3.1E+01	99866	1.1E+01	89752
8	0.0E+00	21647	1.0E+01	94857	3.4E+01	98112	1.4E+01	97675
9	0.0E+00	20976	9.0E+00	84773	3.2E+01	97004	6.0E+00	93011
10	0.0E+00	22102	1.1E+01	99416	2.9E+01	97452	1.0E+01	96110
11	0.0E+00	22442	1.4E+01	98969	2.8E+01	99284	1.0E+01	83658
12	0.0E+00	20017	1.3E+01	99173	3.5E+01	85217	1.2E+01	97762
13	0.0E+00	23083	9.0E+00	97035	3.1E+01	99565	1.5E+01	88237
14	0.0E+00	18376	1.0E+01	98811	3.1E+01	89842	1.6E+01	91411
15	0.0E+00	24595	9.0E+00	96624	3.1E+01	91759	9.0E+00	89680
16	0.0E+00	20493	8.0E+00	99186	3.5E+01	92329	1.3E+01	93811
17	0.0E+00	19287	9.0E+00	95448	3.5E+01	88478	1.4E+01	99298
18	0.0E+00	22211	8.0E+00	100000	3.4E+01	98424	7.0E+00	99252
19	0.0E+00	24338	1.2E+01	80218	3.1E+01	98899	8.0E+00	99600
20	0.0E+00	21768	6.0E+00	96806	3.4E+01	92990	8.0E+00	95507
21	0.0E+00	20406	1.1E+01	99268	3.2E+01	98731	1.5E+01	95866
22	0.0E+00	22986	1.3E+01	60031	3.0E+01	98987	1.4E+01	93647
23	0.0E+00	25199	9.0E+00	98144	3.2E+01	86930	1.1E+01	86010
24	0.0E+00	18735	9.0E+00	90100	3.7E+01	92321	1.3E+01	99969
25	0.0E+00	21111	8.0E+00	71001	3.0E+01	96093	1.2E+01	91482
26	0.0E+00	20796	8.0E+00	99102	3.5E+01	99546	1.0E+01	93419
27	0.0E+00	20959	1.0E+01	85368	2.8E+01	93229	1.1E+01	93362
28	0.0E+00	24399	1.3E+01	97787	3.6E+01	98793	1.5E+01	86453
29	0.0E+00	21003	1.0E+01	92059	3.3E+01	99844	1.8E+01	99862
30	0.0E+00	21371	9.0E+00	92228	3.6E+01	81027	1.2E+01	97723
MEDIA	**0.0E+00**	**21685**	**9.8E+00**	**93101**	**3.3E+01**	**94420**	**1.2E+01**	**94041**

Tabla 6.15: *Resultados obtenidos en la función One-Max (f_{One})*

Función One-Max (f_{One})								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	0.0E+00	95367	1.0E+01	96640	2.0E+00	64622	0.0E+00	23364
2	0.0E+00	90842	5.0E+00	98057	0.0E+00	63925	0.0E+00	22570
3	0.0E+00	93779	7.0E+00	96970	0.0E+00	69786	0.0E+00	20241
4	0.0E+00	92556	8.0E+00	78322	2.0E+00	69813	0.0E+00	24099
5	0.0E+00	95103	1.0E+01	87511	0.0E+00	67929	0.0E+00	20160
6	0.0E+00	97304	7.0E+00	89801	0.0E+00	66509	0.0E+00	22111
7	0.0E+00	95450	7.0E+00	98674	0.0E+00	64870	0.0E+00	49069
8	0.0E+00	94610	9.0E+00	89817	1.0E+00	65374	0.0E+00	20887
9	0.0E+00	97895	6.0E+00	86352	0.0E+00	62616	0.0E+00	21016
10	0.0E+00	94354	7.0E+00	94500	0.0E+00	68645	0.0E+00	22309
11	0.0E+00	94998	7.0E+00	98859	0.0E+00	65973	0.0E+00	26093
12	0.0E+00	97622	6.0E+00	94072	0.0E+00	66072	0.0E+00	51648
13	0.0E+00	95279	6.0E+00	75304	0.0E+00	67990	0.0E+00	87115
14	0.0E+00	96354	7.0E+00	93248	0.0E+00	64054	0.0E+00	28761
15	0.0E+00	97594	6.0E+00	88191	2.0E+00	68726	0.0E+00	21903
16	0.0E+00	93183	5.0E+00	99862	2.0E+00	69101	0.0E+00	19489
17	0.0E+00	90694	3.0E+00	98995	0.0E+00	62515	0.0E+00	20666
18	0.0E+00	94464	4.0E+00	96807	0.0E+00	63493	0.0E+00	20931
19	0.0E+00	96138	7.0E+00	96970	0.0E+00	64880	0.0E+00	25255
20	0.0E+00	98347	4.0E+00	98701	0.0E+00	66092	0.0E+00	20888
21	0.0E+00	97869	7.0E+00	91252	0.0E+00	66150	0.0E+00	21555
22	0.0E+00	97990	8.0E+00	93775	0.0E+00	64464	0.0E+00	18360
23	0.0E+00	97925	4.0E+00	99017	0.0E+00	60400	0.0E+00	22192
24	0.0E+00	96686	1.0E+01	96691	0.0E+00	69263	0.0E+00	22844
25	0.0E+00	94319	1.0E+01	97758	2.0E+00	68938	0.0E+00	21666
26	0.0E+00	95795	7.0E+00	98435	0.0E+00	64340	0.0E+00	22137
27	0.0E+00	97221	5.0E+00	97619	0.0E+00	72613	0.0E+00	18284
28	1.0E+00	91786	8.0E+00	97405	1.0E+00	67264	0.0E+00	18790
29	0.0E+00	93905	6.0E+00	98477	0.0E+00	68071	0.0E+00	87664
30	1.0E+00	92426	7.0E+00	99287	0.0E+00	66276	0.0E+00	77845
MEDIA	**6.7E-02**	**95262**	**6.8E+00**	**94246**	**4.0E-01**	**66359**	**0.0E+00**	**29997**

Tabla 6.16: *Resultados obtenidos en la función Fully deceptive order-3 (f_{Dec})*

Función Fully deceptive order-3 (f_{Dec})								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	1.6E+01	1405	2.0E+00	84841	0.0E+00	22986	1.4E+01	24384
2	1.2E+01	1144	6.0E+00	78741	0.0E+00	29914	2.0E+00	81675
3	1.4E+01	3008	2.0E+00	78078	0.0E+00	23807	4.0E+00	50643
4	1.4E+01	2394	4.0E+00	90780	0.0E+00	23329	2.0E+00	88151
5	1.2E+01	1144	4.0E+00	41352	0.0E+00	20906	0.0E+00	92250
6	1.0E+01	960	4.0E+00	95394	0.0E+00	22535	6.0E+00	2350
7	1.2E+01	4581	4.0E+00	71699	0.0E+00	24870	4.0E+00	49101
8	1.4E+01	3798	4.0E+00	74051	0.0E+00	24302	4.0E+00	87640
9	8.0E+00	1718	4.0E+00	81780	0.0E+00	20460	2.0E+00	51029
10	1.4E+01	1720	4.0E+00	59864	0.0E+00	21478	6.0E+00	53932
11	8.0E+00	2151	4.0E+00	16562	0.0E+00	19566	4.0E+00	41971
12	1.0E+01	1595	4.0E+00	39224	0.0E+00	20312	2.0E+00	99014
13	6.0E+00	1750	6.0E+00	50341	0.0E+00	21244	1.0E+01	93678
14	1.0E+01	1820	1.0E+01	84761	0.0E+00	22772	6.0E+00	69143
15	8.0E+00	2467	1.2E+01	1360	0.0E+00	18154	0.0E+00	51570
16	8.0E+00	1798	8.0E+00	2805	0.0E+00	22826	8.0E+00	61781
17	6.0E+00	2006	4.0E+00	76133	0.0E+00	17945	2.0E+00	85579
18	1.0E+01	1434	2.0E+00	85447	0.0E+00	20228	4.0E+00	73004
19	8.0E+00	1603	8.0E+00	2721	0.0E+00	20976	2.0E+00	67472
20	1.2E+01	1325	6.0E+00	2052	0.0E+00	20624	2.0E+00	88173
21	1.0E+01	1786	2.0E+00	45126	0.0E+00	24742	2.0E+00	91066
22	6.0E+00	2272	8.0E+00	98594	0.0E+00	27119	0.0E+00	86123
23	1.0E+01	1259	1.2E+01	1872	0.0E+00	18880	0.0E+00	75775
24	8.0E+00	837	8.0E+00	1543	0.0E+00	21974	2.0E+00	38000
25	8.0E+00	1638	6.0E+00	63767	0.0E+00	18992	2.0E+00	25562
26	1.6E+01	2444	6.0E+00	78741	0.0E+00	23405	2.0E+00	63961
27	4.0E+00	1591	8.0E+00	36641	0.0E+00	24399	4.0E+00	95090
28	2.0E+00	2126	6.0E+00	45076	0.0E+00	21977	0.0E+00	39102
29	1.0E+01	1809	2.0E+00	96741	0.0E+00	20902	8.0E+00	60737
30	1.2E+01	2209	4.0E+00	51799	0.0E+00	27255	4.0E+00	46459
MEDIA	**9.9E+00**	**1926**	**5.5E+00**	**54596**	**0.0E+00**	**22296**	**3.6E+00**	**64480**

Tabla 6.17: *Resultados obtenidos en la función Fully deceptive order-3 (f_{Dec})*

Función Fully deceptive order-3 (f_{Dec})								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	0.0E+00	13123	1.0E+01	2634	1.0E+01	29879	0.0E+00	77296
2	4.0E+00	25517	1.0E+01	38859	4.0E+00	38677	1.0E+01	99140
3	6.0E+00	41897	2.0E+00	34333	4.0E+00	22954	0.0E+00	80379
4	2.0E+00	66939	6.0E+00	2473	4.0E+00	38905	0.0E+00	30258
5	0.0E+00	26097	8.0E+00	39463	6.0E+00	29795	2.0E+00	74403
6	2.0E+00	33248	6.0E+00	58738	1.4E+01	31975	0.0E+00	92061
7	0.0E+00	38657	1.0E+01	1557	6.0E+00	30502	0.0E+00	58732
8	0.0E+00	10919	1.2E+01	88146	4.0E+00	38278	0.0E+00	68391
9	2.0E+00	29246	1.0E+01	99483	6.0E+00	38303	0.0E+00	57521
10	4.0E+00	35871	8.0E+00	2041	6.0E+00	34638	0.0E+00	57256
11	6.0E+00	43660	1.6E+01	2122	8.0E+00	48387	4.0E+00	15745
12	0.0E+00	28635	1.0E+01	3798	8.0E+00	31086	2.0E+00	47843
13	6.0E+00	49245	1.0E+01	2581	4.0E+00	37308	0.0E+00	56673
14	0.0E+00	23327	8.0E+00	82287	8.0E+00	32049	0.0E+00	95847
15	0.0E+00	20015	8.0E+00	91766	8.0E+00	57567	0.0E+00	76958
16	4.0E+00	43883	8.0E+00	1816	1.0E+01	26713	0.0E+00	74944
17	2.0E+00	35253	8.0E+00	64896	6.0E+00	31690	0.0E+00	47186
18	6.0E+00	36743	1.4E+01	44000	6.0E+00	43732	0.0E+00	77311
19	0.0E+00	71938	6.0E+00	21449	1.2E+01	24426	0.0E+00	99532
20	2.0E+00	33281	8.0E+00	2802	4.0E+00	29940	4.0E+00	84823
21	0.0E+00	25873	4.0E+00	67809	1.2E+01	27410	2.0E+00	20095
22	2.0E+00	59246	6.0E+00	61761	8.0E+00	28641	0.0E+00	29004
23	0.0E+00	57466	4.0E+00	16293	6.0E+00	28451	0.0E+00	79586
24	2.0E+00	25998	4.0E+00	3099	6.0E+00	35751	2.0E+00	76650
25	2.0E+00	35737	2.0E+00	1540	2.0E+00	39896	0.0E+00	44316
26	2.0E+00	58912	1.2E+01	77174	4.0E+00	35262	2.0E+00	86116
27	0.0E+00	19064	8.0E+00	60345	1.2E+01	31784	0.0E+00	30967
28	2.0E+00	23051	1.0E+01	2236	6.0E+00	25389	0.0E+00	71144
29	2.0E+00	50628	8.0E+00	2416	4.0E+00	27665	0.0E+00	47186
30	0.0E+00	29484	8.0E+00	1435	4.0E+00	27055	0.0E+00	87732
MEDIA	**1.9E+00**	**36432**	**8.1E+00**	**32645**	**6.7E+00**	**33470**	**9.3E-01**	**64836**

Tabla 6.18: *Resultados obtenidos en la función Royal Road (f_{RR})*

Función Royal Road (f_{RR})								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	5.6E+01	82527	3.2E+01	75363	7.2E+01	53061	4.0E+01	60987
2	4.8E+01	93974	3.2E+01	98252	9.6E+01	36790	3.2E+01	69146
3	6.4E+01	91876	3.2E+01	79539	8.0E+01	31976	4.8E+01	88991
4	1.6E+01	94547	2.4E+01	95844	6.4E+01	97662	4.0E+01	68929
5	4.0E+01	99593	4.8E+01	97942	8.0E+01	78437	2.4E+01	66488
6	3.2E+01	72066	8.0E+00	97157	6.4E+01	86591	2.4E+01	81928
7	3.2E+01	81042	2.4E+01	89789	8.0E+01	45275	3.2E+01	87143
8	2.4E+01	87310	2.4E+01	98665	5.6E+01	77556	3.2E+01	86534
9	4.8E+01	89936	2.4E+01	93418	7.2E+01	55956	8.0E+00	84453
10	6.4E+01	75394	4.0E+01	33300	4.8E+01	94296	2.4E+01	96966
11	5.6E+01	96279	2.4E+01	67467	6.4E+01	89030	4.0E+01	58348
12	4.0E+01	97255	2.4E+01	95420	8.0E+01	85827	3.2E+01	72564
13	2.4E+01	98505	2.4E+01	82593	8.0E+01	87504	4.0E+01	78961
14	8.0E+00	95315	4.0E+01	89605	7.2E+01	86850	4.0E+01	55400
15	6.4E+01	99356	2.4E+01	61576	8.8E+01	31476	5.6E+01	36439
16	3.2E+01	96828	2.4E+01	77505	5.6E+01	70196	4.8E+01	49085
17	4.0E+01	87910	3.2E+01	88139	6.4E+01	88055	4.8E+01	54828
18	8.0E+01	94102	4.0E+01	41463	6.4E+01	80972	4.8E+01	81666
19	6.4E+01	89186	3.2E+01	82412	8.0E+01	93765	3.2E+01	76142
20	3.2E+01	99268	4.0E+01	84673	8.0E+01	53069	3.2E+01	46696
21	5.6E+01	79516	4.0E+01	57508	8.0E+01	42153	3.2E+01	93491
22	4.8E+01	86846	3.2E+01	72981	8.0E+01	68738	4.0E+01	90968
23	1.6E+01	91494	2.4E+01	77648	4.0E+01	99813	2.4E+01	76103
24	5.6E+01	66785	1.6E+01	92154	7.2E+01	80036	5.6E+01	42287
25	4.0E+01	69827	1.6E+01	79126	8.0E+01	58009	3.2E+01	71174
26	4.8E+01	98117	4.0E+01	84444	8.0E+01	58877	3.2E+01	60322
27	6.4E+01	59270	1.6E+01	87278	6.4E+01	83776	4.0E+01	37467
28	4.0E+01	90235	8.0E+00	96001	7.2E+01	53068	2.4E+01	43263
29	3.2E+01	90922	3.2E+01	28606	7.2E+01	58969	2.4E+01	85066
30	2.4E+01	70054	2.4E+01	59339	5.6E+01	45261	3.2E+01	70094
MEDIA	**4.3E+01**	**87511**	**2.8E+01**	**78840**	**7.1E+01**	**69101**	**3.5E+01**	**69064**

Tabla 6.19: *Resultados obtenidos en la función Royal Road (f_{RR})*

Función Royal Road (f_{RR})								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.	Eval.	Nº Eval.
1	2.4E+01	85691	4.8E+01	83130	1.6E+01	85149	3.2E+01	93183
2	1.6E+01	96568	4.0E+01	98000	2.4E+01	86310	4.8E+01	88519
3	2.4E+01	85691	4.8E+01	94999	2.4E+01	73322	1.6E+01	94062
4	3.2E+01	97743	4.0E+01	80827	2.4E+01	85891	3.2E+01	84172
5	4.8E+01	77400	5.6E+01	93686	1.6E+01	89085	3.2E+01	78373
6	1.6E+01	96093	4.8E+01	78729	4.0E+01	82441	3.2E+01	93183
7	2.4E+01	94151	4.8E+01	83183	1.6E+01	82303	3.2E+01	95189
8	8.0E+00	93438	2.4E+01	89026	3.2E+01	94936	3.2E+01	96998
9	1.6E+01	96093	4.8E+01	89440	2.4E+01	96311	3.2E+01	94912
10	4.0E+01	96386	4.8E+01	88867	8.0E+00	71977	4.8E+01	35100
11	3.2E+01	82009	4.8E+01	88281	2.4E+01	83322	4.8E+01	95529
12	2.4E+01	86404	4.0E+01	80811	3.2E+01	83421	2.4E+01	86541
13	0.0E+00	98430	5.6E+01	94229	2.4E+01	83718	4.8E+01	99989
14	2.4E+01	99852	4.8E+01	92379	2.4E+01	68895	2.4E+01	62660
15	3.2E+01	92839	4.8E+01	69811	3.2E+01	84964	3.2E+01	89535
16	2.4E+01	93524	3.2E+01	96184	2.4E+01	74362	3.2E+01	87612
17	1.6E+01	90155	3.2E+01	86418	3.2E+01	98217	4.8E+01	79114
18	4.0E+01	98735	6.4E+01	80912	3.2E+01	92674	4.0E+01	63504
19	1.6E+01	95894	3.2E+01	88059	1.6E+01	85674	3.2E+01	73425
20	4.0E+01	85983	4.0E+01	67439	1.6E+01	94735	3.2E+01	86855
21	4.0E+01	83244	2.4E+01	86053	1.6E+01	89417	4.0E+01	86368
22	3.2E+01	74079	4.0E+01	84955	8.0E+00	75854	3.2E+01	91309
23	1.6E+01	94809	5.6E+01	84816	2.4E+01	92345	4.0E+01	91511
24	1.6E+01	91648	6.4E+01	54369	3.2E+01	76821	4.0E+01	44090
25	2.4E+01	85315	8.0E+01	89924	1.6E+01	87513	1.6E+01	65540
26	2.4E+01	86404	6.4E+01	75402	2.4E+01	80315	4.0E+01	85428
27	2.4E+01	90913	4.8E+01	91233	8.0E+00	88871	3.2E+01	94519
28	2.4E+01	94415	1.6E+01	87325	2.4E+01	84657	4.8E+01	60622
29	4.8E+01	79941	6.4E+01	96454	2.4E+01	79990	4.0E+01	80566
30	1.6E+01	96093	3.2E+01	91513	1.6E+01	85219	8.0E+00	90823
MEDIA	**2.5E+01**	**90665**	**4.6E+01**	**85548**	**2.2E+01**	**84624**	**3.4E+01**	**82308**

6.5.4. Comparación. Análisis estadísticos

Por último, para comparar los resultados obtenidos con AG_SGA frente al resto de algoritmos se han realizado diversos tests estadísticos. Para determinar la significación se ha utilizado el valor estándar de p-valor$<$0.05.

Previamente, se ha llevado a cabo la prueba de Kolmogorov-Smirnov para contrastar la normalidad de los datos. De esta forma hemos podido conocer si era posible aplicar el test t-Student. En la prueba de Kolmogorov-Smirnov se ha denominado hipótesis nula (H_0) a la hipótesis que se supone cierta de partida e hipótesis alternativa (H_1) a la que reemplazará a la hipótesis nula cuando ésta sea rechazada. La hipótesis que ha sido contrastada es:

- H_0: los datos siguen una distribución normal
- H_1: los datos no siguen una distribución normal

Los resulados de la prueba, para cada grupo de datos de cada función, se muestran en las figuras 6.12, 6.13, 6.14, 6.15, 6.16 y 6.17.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	2.4E-010	9.7E-008	9.0E-006	1.8E-007	3.9E-010
	Desviación típica	.000[c]	1.26E-007	1.13E-005	2.55E-007	3.02E-010
Diferencias más extremas	Absoluta		.292	.280	.292	.385
	Positiva		.292	.280	.292	.385
	Negativa		-.228	-.222	-.246	-.315
Z de Kolmogorov-Smirnov			1.601	1.536	1.601	2.109
Sig. asintót. (bilateral)			.012	.018	.012	.000

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

c. La distribución no tiene varianza para esta variable. No es posible realizar la prueba de Kolmogorov-Smirnov para una muestra.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	2.4E-010	2.7E-010	2.4E-010	2.4E-010
	Desviación típica	.000[c]	8.74E-011	.000[c]	.000[c]
Diferencias más extremas	Absoluta		.508		
	Positiva		.508		
	Negativa		-.359		
Z de Kolmogorov-Smirnov			2.780		
Sig. asintót. (bilateral)			.000		

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

c. La distribución no tiene varianza para esta variable. No es posible realizar la prueba de Kolmogorov-Smirnov para una muestra.

Figura 6.12: *Resultados de la prueba Kolmogorov-Smirnov en la función* f_{Sph}.

Como se aprecia, en la función f_{Sph} no ha sido posible aplicar la prueba en los grupos de datos de los algoritmos AG_EST1, AG_IL, AG_FAMP y AG_SGA. Esto ha sido debido a que las distribuciones de dichos datos son constantes y no tienen varianza, ya que el resultado obtenido por cada algoritmo siembre ha sido el mismo: la mejor solución posible. En el resto de algoritmos, p-valor[1]<0.05, los datos no siguen una distribución normal y la hipótesis nula ha sido rechazada. Por tanto, se ha realizado la prueba no paramétrica de Mann-Whitney.

[1]p-valor se identifica como *Sig. asintót. (bilateral)* en la prueba de Kolmogorov-Smirnov

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	.121	.0028	2.12E-007	.00096	.00047
	Desviación típica	1.92E-001	5.41E-003	1.15E-006	2.65E-003	1.36E-003
Diferencias más extremas	Absoluta	.266	.304	.539	.440	.426
	Positiva	.220	.294	.539	.440	.426
	Negativa	-.266	-.304	-.428	-.359	-.366
Z de Kolmogorov-Smirnov		1.457	1.665	2.953	2.412	2.334
Sig. asintót. (bilateral)		.029	.008	.000	.000	.000

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	.102	.023	.00019	2.0E-009
	Desviación típica	.1690	.0378	.0007	5.8E-010
Diferencias más extremas	Absoluta	.292	.334	.502	.281
	Positiva	.292	.334	.502	.281
	Negativa	-.273	-.275	-.398	-.216
Z de Kolmogorov-Smirnov		1.601	1.832	2.749	1.538
Sig. asintót. (bilateral)		.012	.002	.000	.018

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Figura 6.13: *Resultados de la prueba Kolmogorov-Smirnov en la función* f_{Ros}.

En la función f_{Ros}, todas las pruebas han proporcionado un p-valor<0.05, por lo que la hipótesis nula ha tenido que ser rechazada. También, en este caso, se ha llevado a cabo la prueba de Mann-Whitney.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	7.6	8.9E-005	.079	.00013	.033
	Desviación típica	3.6981	.0001	.2540	.0003	.1817
Diferencias más extremas	Absoluta	.140	.275	.482	.327	.539
	Positiva	.140	.275	.482	.290	.539
	Negativa	-.071	-.244	-.379	-.327	-.428
Z de Kolmogorov-Smirnov		.765	1.504	2.639	1.790	2.953
Sig. asintót. (bilateral)		.601	.022	.000	.003	.000

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	2.6	.4	5.0E-008	4.8E-008
	Desviación típica	2.1702	1.1561	9.4E-009	.00[c]
Diferencias más extremas	Absoluta	.167	.466	.537	
	Positiva	.167	.466	.537	
	Negativa	-.112	-.365	-.396	
Z de Kolmogorov-Smirnov		.914	2.554	2.941	
Sig. asintót. (bilateral)		.374	.000	.000	

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

c. La distribución no tiene varianza para esta variable. No es posible realizar la prueba de Kolmogorov-Smirnov para una muestra.

Figura 6.14: *Resultados de la prueba Kolmogorov-Smirnov en la función* f_{Ras}.

En las funciones f_{Ras} y f_{One} no se ha podido realizar la prueba con los datos proporcionados por el algoritmo AG_SGA, ya que las distribuciones de dichos datos son constantes y no tienen varianza, debido a que el algoritmo ha hallado en todos los experimentos al mejor individuo posible. Por tanto, independientemente del resultado en el resto de grupos, se ha tenido que aplicar la prueba de Mann-Whitney.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	.0	9.8	33.1	11.7	.067
	Desviación típica	.0000[c]	2.1827	3.0483	2.9117	.2537
Diferencias más extremas	Absoluta		.182	.158	.136	.537
	Positiva		.182	.158	.067	.537
	Negativa		-.134	-.097	-.136	-.396
Z de Kolmogorov-Smirnov			.997	.865	.748	2.941
Sig. asintót. (bilateral)			.273	.442	.631	.000

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

c. La distribución no tiene varianza para esta variable. No es posible realizar la prueba de Kolmogorov-Smirnov para una muestra.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	6.77	.4	.0	.0
	Desviación típica	1.8696	.7701	.0000[c]	.0000[c]
Diferencias más extremas	Absoluta	.184	.465		
	Positiva	.184	.465		
	Negativa	-.150	-.302		
Z de Kolmogorov-Smirnov		1.006	2.546		
Sig. asintót. (bilateral)		.264	.000		

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

c. La distribución no tiene varianza para esta variable. No es posible realizar la prueba de Kolmogorov-Smirnov para una muestra.

Figura 6.15: *Resultados de la prueba Kolmogorov-Smirnov en la función* f_{One}.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	9.9	5.5	.0	3.6	1.9
	Desviación típica	3.383	2.776	.000[c]	3.212	2.067
Diferencias más extremas	Absoluta	.125	.235		.224	.254
	Positiva	.125	.235		.224	.254
	Negativa	-.117	-.132		-.143	-.175
Z de Kolmogorov-Smirnov		.687	1.286		1.228	1.390
Sig. asintót. (bilateral)		.732	.073		.098	.042

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

c. La distribución no tiene varianza para esta variable. No es posible realizar la prueba de Kolmogorov-Smirnov para una muestra.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	8.1	6.7	.93	.13
	Desviación típica	3.1919	2.9935	2.0833	.5074
Diferencias más extremas	Absoluta	.183	.230	.406	.537
	Positiva	.150	.230	.406	.537
	Negativa	-.183	-.147	-.327	-.396
Z de Kolmogorov-Smirnov		1.004	1.260	2.225	2.941
Sig. asintót. (bilateral)		.266	.083	.000	.000

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Figura 6.16: *Resultados de la prueba Kolmogorov-Smirnov en la función* f_{Dec}.

En f_{Dec} se ha obtenido un p-valor<0.05 en los grupos de datos de AG_DET, AG_FAMP y AG_SGA, por lo que también se ha rechazado la hipótesis nula. Por último, en la función f_{RR}, p-valor>0.05 en todos los datos de los algoritmos, por lo que se acepta la hipótesis nula y, únicamente en este caso, ha sido posible realizar el t-Student.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	42.9	28.0	71.2	35.2	25.3
	Desviación típica	17.386	9.798	12.133	10.630	11.330
Diferencias más extremas	Absoluta	.107	.192	.199	.185	.214
	Positiva	.102	.192	.167	.185	.214
	Negativa	-.107	-.175	-.199	-.148	-.138
Z de Kolmogorov-Smirnov		.587	1.050	1.091	1.013	1.169
Sig. asintót. (bilateral)		.881	.220	.185	.256	.130

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	45.9	22.4	34.4	23.7
	Desviación típica	13.93	7.97	10.11	10.40
Diferencias más extremas	Absoluta	.172	.213	.239	.172
	Positiva	.172	.187	.194	.172
	Negativa	-.161	-.213	-.239	-.120
Z de Kolmogorov-Smirnov		.945	1.166	1.312	.940
Sig. asintót. (bilateral)		.334	.132	.064	.340

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Figura 6.17: *Resultados de la prueba Kolmogorov-Smirnov en la función* f_{RR}.

Una vez finalidadas las pruebas de normalidad se ha tenido conocimiento sobre el test que debía ser aplicado en cada caso. La tabla 6.20 muestra el resultado de cada uno de los tests, al comparar el comportamiento de AG_SGA con el resto de algoritmos. Se ha denotado con un signo + cuando ha existido una mejora en el comportamiento de AG_SGA con respecto al algoritmo, un signo - si no ha existido mejora y, un signo $\approx$ cuando no han existido diferencias significativas. Al igual que en la tabla 6.7, la primera columna identifica el algoritmo utilizado. Posteriormente, para cada función, existen tres columnas identificadas como ME, MNE y NO. La columna ME muestra la media de las evaluaciones de los mejores individuos obtenidos al final de cada ejecución del algoritmo, la columna MNE muestra la media del número de evaluaciones a partir de la cual no ha existido una mejora en la solución encontrada y, por último, la columna NO muestra el número de veces que el AG ha encontrado al mejor individuo posible.

Tabla 6.20: *Resultados de los tests estadísticos en las funciones de prueba.*

	f_{Sph}			f_{Ros}			f_{Ras}		
Algoritmo	ME	NE	O	ME	NE	O	ME	NE	O
AG_EST1	2.4E-10 ≈	34350 +	30	1.2E-01 +	99885 +	0	7.6E+00 +	74544 +	0
AG_EST2	9.7E-08 +	98139 +	0	2.8E-03 +	88366 +	4	8.9E-05 +	98511 +	0
AG_EST3	9.0E-06 +	98326 +	0	2.1E-07 ≈	50088 ≈	29	7.9E-02 +	99336 +	0
AG_RAN	1.8E-07 +	97553 +	0	9.6E-04 ≈	84255 +	13	1.3E-04 +	98600 +	0
AG_DET	3.8E-10 +	97508 +	21	4.7E-04 ≈	56256 ≈	10	3.3E-02 +	98435 +	21
AG_IL	2.4E-10 ≈	61502 +	30	1.0E-01 +	99772 +	0	2.6E+00 +	93311 +	2
AG_SELF	2.7E-10 +	85631 +	26	2.3E-02 +	96908 +	0	4.0E-01 +	93782 +	1
AG_FAMP	2.4E-10 ≈	40255 +	30	1.9E-04 ≈	76056 +	19	5.0E-08 ≈	70484 +	28
AG_SGA	2.4E-10	12189	30	2.0E-09	58265	10	4.8E-08	38886	30
	f_{One}			f_{Dec}			f_{RR}		
Alg.	ME	NE	O	ME	NE	O	ME	NE	O
AG_EST1	0.0E+00 ≈	21685 +	30	9.9E+00 +	1926 -	0	4.3E+01 +	87511 ≈	0
AG_EST2	9.8E+00 +	93101 +	0	5.5E+00 +	54596 ≈	0	2.8E+01 ≈	78840 -	0
AG_EST3	3.3E+01 +	94420 +	0	0.0E+00 ≈	22296 -	30	7.1E+01 +	69101 -	0
AG_RAN	1.2E+01 +	94041 +	0	3.6E+00 +	64480 ≈	5	3.5E+01 +	69064 -	0
AG_DET	6.7E-02 ≈	95262 +	28	1.9E+00 +	36432 +	12	2.5E+01 ≈	90665 ≈	1
AG_IL	6.8E+00 +	94246 +	0	8.1E+00 +	32645 +	0	4.6E+01 +	85548 ≈	0
AG_SELF	4.0E-01 +	66359 +	23	6.7E+00 +	33470 +	0	2.2E+01 ≈	84624 ≈	0
AG_FAMP	0.0E+00 ≈	29997 +	30	9.3E-01 +	64836 ≈	22	3.4E+01 +	82308 +	0
AG_SGA	0.0E+00	10276	30	1.3E-01	59272	28	2.4E+01	89052	0

A partir de los resultados obtenidos podemos destacar los siguientes aspectos:

1. En los algoritmos con unos parámetros estáticos, AG_EST1, AG_EST2 y AG_EST3, se observa que los mejores resultados no han sido obtenidos siempre con el mismo algoritmo. AG_EST1 es el mejor en las funciones f_{Sph} y f_{One}, AG_EST2 en las funciones f_{Ras} y f_{RR}, y AG_EST3 en f_{Ros} y f_{Dec}. Por tanto, para alcanzar los mejores resultados ha sido necesario utilizar una P_m fija distinta en cada una de las funciones.

2. Al comparar AG_SGA vs. AG_EST1, AG_EST2 y AG_EST3 se aprecia que, en la mayoria de las funciones, no han existido diferencias significativas en la media de las evaluaciones de los mejores individuos (ME) y, únicamente, en la función f_{Ras} el comportamiento de AG_SGA ha sido mucho mejor. Por

tanto, ninguno de los algoritmos con unos parámetros estáticos ha sido capaz de superar a AG_SGA, que ha igualado o mejorado el comportamiento del mejor algoritmo en cada una de las funciones.

3. Al contrastar AG_SGA vs. AG_RAN se confirma que el algoritmo AG_SGA ha sido mucho mejor ya que, en la mayoria de las funciones (excepto en f_{Ros}), han existido diferencias significativas, tanto en la ME como en la media del número de evaluaciones a partir de la cual no ha existido una mejora en la solución encontrada (MNE).

4. Al confrontar AG_SGA vs. AG_DET se verifica que AG_SGA ha mejorado a AG_DET en la ME y en la MNE de las funciones f_{Sph}, f_{Ras} y f_{Dec}. En la función f_{One} se han obtenido iguales resultados en la ME, sin embargo, han existido diferencias significativas en la MNE, lo cual indica que la velocidad en alcanzar el mismo resultado es mucho mayor en AG_SGA. En las otras dos funciones no han existido diferencias.

5. Al cotejar AG_SGA vs. AG_IL y AG_SELF se ha constatado que, en la mayoría de las funciones, AG_SGA ha obtenido mejores resultados tanto en la ME como en la MNE. AG_SELF, únicamente, ha igualado el comportamiento de AG_SGA en f_{RR}, y AG_IL en f_{Sph}; aunque en esta última función el algoritmo AG_SGA ha sido mucho más rápido en alcanzar el mismo resultado.

6. Al parangonar AG_SGA vs. AG_FAMP se ha comprobado que AG_FAMP ha sido el algoritmo que ha tenido un comportamiento más parecido a AG_SGA. En las funciones f_{Sph}, f_{Ros}, f_{Ras} y f_{One} no han existido diferencias significativas en la ME. Sin embargo, AG_SGA ha sido mucho más rápido en lograr los mismos resultados en las cuatro funciones ya que existen diferencias significativas en la MNE. En las funciones f_{Dec} y f_{RR}, AG_SGA ha mejorado a AG_FAMP en la ME y en la función f_{RR} también en la MNE.

7. En ningún caso, el comportamiento de AG_SGA ha sido peor que el resto de algoritmos, no existiendo ninguno que se haya comportado como él en las 6 funciones. Así, AG_SGA se ha comportado mejor en 4 de las 6 funciones que AG_EST1; en 5 de las 6 que AG_EST2; en 4 que AG_EST3; en 5 que AG_RAN; en 3 que AG_DET; en 5 que AG_IL; en 5 que AG_SELF y en 2 que AG_FAMP.

Por tanto, a la vista de estos resultados, podemos afirmar que el comportamiento del AG_SGA ha sido muy robusto, siendo el algoritmo más efectivo y versátil en la optimización de las seis funciones de prueba.

Las siguientes figuras muestran los resultados de las pruebas estadísticas Mann-Whitney[2] y t-Student[3] en cada caso.

Resultados de las pruebas estadísticas en la función f_{Sph}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	30.50	915.00
	2.00	30	30.50	915.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	450.000
W de Wilcoxon	915.000
Z	.000
Sig. asintót. (bilateral)	1.000

a. Variable de agrupación: grupo

Figura 6.18: *Prueba Mann-Whitney AG_SGA vs. AG_EST1 en* f_{Sph}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-7.112
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.19: *Prueba Mann-Whitney AG_SGA vs. AG_EST2 en* f_{Sph}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-7.112
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.20: *Prueba Mann-Whitney AG_SGA vs. AG_EST3 en* f_{Sph}.

[2]p-valor se identifica como *Sig. asintót. (bilateral)* en la prueba Mann-Whitney
[3]p-valor se identifica como *Sig.(bilateral)* en la prueba t-Student

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-7.112
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.21: *Prueba Mann-Whitney AG_SGA vs. AG_RAN en* f_{Sph}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	26.00	780.00
	2.00	30	35.00	1050.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	315.000
W de Wilcoxon	780.000
Z	-3.214
Sig. asintót. (bilateral)	.001

a. Variable de agrupación: grupo

Figura 6.22: *Prueba Mann-Whitney AG_SGA vs. AG_DET en* f_{Sph}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	30.50	915.00
	2.00	30	30.50	915.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	450.000
W de Wilcoxon	915.000
Z	.000
Sig. asintót. (bilateral)	1.000

a. Variable de agrupación: grupo

Figura 6.23: *Prueba Mann-Whitney AG_SGA vs. AG_IL en* f_{Sph}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	28.50	855.00
	2.00	30	32.50	975.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	390.000
W de Wilcoxon	855.000
Z	-2.052
Sig. asintót. (bilateral)	.040

a. Variable de agrupación: grupo

Figura 6.24: *Prueba Mann-Whitney AG_SGA vs. AG_SELF en* f_{Sph}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	30.50	915.00
	2.00	30	30.50	915.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	450.000
W de Wilcoxon	915.000
Z	.000
Sig. asintót. (bilateral)	1.000

a. Variable de agrupación: grupo

Figura 6.25: *Prueba Mann-Whitney AG_SGA vs. AG_FAMP en* f_{Sph}.

Resultados de las pruebas estadísticas en la función f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-6.678
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.26: *Prueba Mann-Whitney AG_SGA vs. AG_EST1 en* f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	19.80	594.00
	2.00	30	41.20	1236.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	129.000
W de Wilcoxon	594.000
Z	-4.786
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.27: *Prueba Mann-Whitney AG_SGA vs. AG_EST2 en* f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	34.08	1022.50
	2.00	30	26.92	807.50
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	342.500
W de Wilcoxon	807.500
Z	-1.678
Sig. asintót. (bilateral)	.093

a. Variable de agrupación: grupo

Figura 6.28: *Prueba Mann-Whitney AG_SGA vs. AG_EST3 en* f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	28.80	864.00
	2.00	30	32.20	966.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	399.000
W de Wilcoxon	864.000
Z	-.779
Sig. asintót. (bilateral)	.436

a. Variable de agrupación: grupo

Figura 6.29: *Prueba Mann-Whitney AG_SGA vs. AG_RAN en* f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	28.90	867.00
	2.00	30	32.10	963.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	402.000
W de Wilcoxon	867.000
Z	-.726
Sig. asintót. (bilateral)	.468

a. Variable de agrupación: grupo

Figura 6.30: *Prueba Mann-Whitney AG_SGA vs. AG_DET en* f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-6.677
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.31: *Prueba Mann-Whitney AG_SGA vs. AG_IL en* f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.55	466.50
	2.00	30	45.45	1363.50
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	1.500
W de Wilcoxon	466.500
Z	-6.655
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.32: *Prueba Mann-Whitney AG_SGA vs. AG_SELF en* f_{Ros}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	33.03	991.00
	2.00	30	27.97	839.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	374.000
W de Wilcoxon	839.000
Z	-1.197
Sig. asintót. (bilateral)	.231

a. Variable de agrupación: grupo

Figura 6.33: *Prueba Mann-Whitney AG_SGA vs. AG_FAMP en* f_{Ros}.

Resultados de las pruebas estadísticas en la función f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-7.118
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.34: *Prueba Mann-Whitney AG_SGA vs. AG_EST1 en* f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-7.112
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.35: *Prueba Mann-Whitney AG_SGA vs. AG_EST2 en* f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-7.112
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.36: *Prueba Mann-Whitney AG_SGA vs. AG_EST3 en* f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.50	465.00
	2.00	30	45.50	1365.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	.000
W de Wilcoxon	465.000
Z	-7.112
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.37: *Prueba Mann-Whitney AG_SGA vs. AG_RAN en* f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	26.00	780.00
	2.00	30	35.00	1050.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	315.000
W de Wilcoxon	780.000
Z	-3.214
Sig. asintót. (bilateral)	.001

a. Variable de agrupación: grupo

Figura 6.38: *Prueba Mann-Whitney AG_SGA vs. AG_DET en* f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	16.50	495.00
	2.00	30	44.50	1335.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	30.000
W de Wilcoxon	495.000
Z	-6.746
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.39: *Prueba Mann-Whitney AG_SGA vs. AG_IL en* f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	16.00	480.00
	2.00	30	45.00	1350.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	15.000
W de Wilcoxon	480.000
Z	-6.930
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.40: *Prueba Mann-Whitney AG_SGA vs. AG_SELF en* f_{Ras}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	29.50	885.00
	2.00	30	31.50	945.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	420.000
W de Wilcoxon	885.000
Z	-1.426
Sig. asintót. (bilateral)	.154

a. Variable de agrupación: grupo

Figura 6.41: *Prueba Mann-Whitney AG_SGA vs. AG_FAMP en* f_{Ras}.

Resultados de las pruebas estadísticas en la función f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	30.50	915.00
	2.00	30	30.50	915.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	450.000
W de Wilcoxon	915.000
Z	.000
Sig. asintót. (bilateral)	1.000

a. Variable de agrupación: grupo

Figura 6.42: *Prueba Mann-Whitney AG_SGA vs. AG_EST1 en* f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	17.00	510.00
	2.00	30	44.00	1320.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	45.000
W de Wilcoxon	510.000
Z	-6.570
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.43: *Prueba Mann-Whitney AG_SGA vs. AG_EST2 en* f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	17.00	510.00
	2.00	30	44.00	1320.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	45.000
W de Wilcoxon	510.000
Z	-6.564
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.44: *Prueba Mann-Whitney AG_SGA vs. AG_EST3 en* f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	17.00	510.00
	2.00	30	44.00	1320.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	45.000
W de Wilcoxon	510.000
Z	-6.562
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.45: *Prueba Mann-Whitney AG_SGA vs. AG_RAN en* f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	29.50	885.00
	2.00	30	31.50	945.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	420.000
W de Wilcoxon	885.000
Z	-1.426
Sig. asintót. (bilateral)	.154

a. Variable de agrupación: grupo

Figura 6.46: *Prueba Mann-Whitney AG_SGA vs. AG_DET en* f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	17.00	510.00
	2.00	30	44.00	1320.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	45.000
W de Wilcoxon	510.000
Z	-6.574
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.47: *Prueba Mann-Whitney AG_SGA vs. AG_IL en* f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	27.50	825.00
	2.00	30	33.50	1005.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	360.000
W de Wilcoxon	825.000
Z	-2.557
Sig. asintót. (bilateral)	.011

a. Variable de agrupación: grupo

Figura 6.48: *Prueba Mann-Whitney AG_SGA vs. AG_SELF en* f_{One}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	30.50	915.00
	2.00	30	30.50	915.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	450.000
W de Wilcoxon	915.000
Z	.000
Sig. asintót. (bilateral)	1.000

a. Variable de agrupación: grupo

Figura 6.49: *Prueba Mann-Whitney AG_SGA vs. AG_FAMP en* f_{One}.

Resultados de las pruebas estadísticas en la función f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.55	466.50
	2.00	30	45.45	1363.50
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	1.500
W de Wilcoxon	466.500
Z	-6.972
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.50: *Prueba Mann-Whitney AG_SGA vs. AG_EST1 en* f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.75	472.50
	2.00	30	45.25	1357.50
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	7.500
W de Wilcoxon	472.500
Z	-6.900
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.51: *Prueba Mann-Whitney AG_SGA vs. AG_EST2 en* f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	32.00	960.00
	2.00	30	29.00	870.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	405.000
W de Wilcoxon	870.000
Z	-1.762
Sig. asintót. (bilateral)	.078

a. Variable de agrupación: grupo

Figura 6.52: *Prueba Mann-Whitney AG_SGA vs. AG_EST3 en* f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	18.80	564.00
	2.00	30	42.20	1266.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	99.000
W de Wilcoxon	564.000
Z	-5.682
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.53: *Prueba Mann-Whitney AG_SGA vs. AG_RAN en* f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	22.65	679.50
	2.00	30	38.35	1150.50
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	214.500
W de Wilcoxon	679.500
Z	-4.125
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.54: *Prueba Mann-Whitney AG_SGA vs. AG_DET en* f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.60	468.00
	2.00	30	45.40	1362.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	3.000
W de Wilcoxon	468.000
Z	-6.959
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.55: *Prueba Mann-Whitney AG_SGA vs. AG_IL en* f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	15.55	466.50
	2.00	30	45.45	1363.50
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	1.500
W de Wilcoxon	466.500
Z	-6.984
Sig. asintót. (bilateral)	.000

a. Variable de agrupación: grupo

Figura 6.56: *Prueba Mann-Whitney AG_SGA vs. AG_SELF en* f_{Dec}.

Rangos

	grupo	N	Rango promedio	Suma de rangos
datos	1.00	30	28.30	849.00
	2.00	30	32.70	981.00
	Total	60		

Estadísticos de contraste[a]

	datos
U de Mann-Whitney	384.000
W de Wilcoxon	849.000
Z	-1.505
Sig. asintót. (bilateral)	.132

a. Variable de agrupación: grupo

Figura 6.57: *Prueba Mann-Whitney AG_SGA vs. AG_FAMP en* f_{Dec}.

Resultados de las pruebas estadísticas en la función f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	4.293E+001	1.7386E+001	3.1742E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	8.574	.005	-5.191	58	.000	-1.9200E+001	3.6984E+000	-2.6603E+001	-1.1797E+001
	No se han asumido varianzas iguales			-5.191	47.386	.000	-1.9200E+001	3.6984E+000	-2.6639E+001	-1.1761E+001

Figura 6.58: *Prueba t-Student AG_SGA vs. AG_EST1 en* f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	2.800E+001	9.7980E+000	1.7889E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.046	.830	-1.636	58	.107	-4.2667E+000	2.6081E+000	-9.4873E+000	9.5400E-001
	No se han asumido varianzas iguales			-1.636	57.798	.107	-4.2667E+000	2.6081E+000	-9.4877E+000	9.5439E-001

Figura 6.59: *Prueba t-Student AG_SGA vs. AG_EST2 en* f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	7.120E+001	1.2133E+001	2.2151E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.505	.480	-16.272	58	.000	-4.7467E+001	2.9170E+000	-5.3306E+001	-4.1628E+001
	No se han asumido varianzas iguales			-16.272	56.668	.000	-4.7467E+001	2.9170E+000	-5.3309E+001	-4.1625E+001

Figura 6.60: *Prueba t-Student AG_SGA vs. AG_EST3 en* f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	3.520E+001	1.0630E+001	1.9407E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.004	.947	-4.224	58	.000	-1.1467E+001	2.7145E+000	-1.6900E+001	-6.0330E+000
	No se han asumido varianzas iguales			-4.224	57.971	.000	-1.1467E+001	2.7145E+000	-1.6900E+001	-6.0329E+000

Figura 6.61: *Prueba t-Student AG_SGA vs. AG_RAN en* f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	2.533E+001	1.1330E+001	2.0686E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.053	.818	-.570	58	.571	-1.6000E+000	2.8073E+000	-7.2195E+000	4.0195E+000
	No se han asumido varianzas iguales			-.570	57.575	.571	-1.6000E+000	2.8073E+000	-7.2204E+000	4.0204E+000

Figura 6.62: *Prueba t-Student AG_SGA vs. AG_DET en* f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	4.587E+001	1.3925E+001	2.5424E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	1.309	.257	-6.976	58	.000	-2.2133E+001	3.1727E+000	-2.8484E+001	-1.5783E+001
	No se han asumido varianzas iguales			-6.976	53.663	.000	-2.2133E+001	3.1727E+000	-2.8495E+001	-1.5772E+001

Figura 6.63: *Prueba t-Student AG_SGA vs. AG_IL en* f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	2.240E+001	7.9724E+000	1.4555E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	2.087	.154	.557	58	.579	1.3333E+000	2.3918E+000	-3.4544E+000	6.1211E+000
	No se han asumido varianzas iguales			.557	54.345	.580	1.3333E+000	2.3918E+000	-3.4613E+000	6.1279E+000

Figura 6.64: *Prueba t-Student AG_SGA vs. AG_SELF en* f_{RR}.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	2.373E+001	1.0395E+001	1.8979E+000
	2.00	30	3.440E+001	1.0108E+001	1.8455E+000

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.158	.692	-4.029	58	.000	-1.0667E+001	2.6473E+000	-1.5966E+001	-5.3675E+000
	No se han asumido varianzas iguales			-4.029	57.955	.000	-1.0667E+001	2.6473E+000	-1.5966E+001	-5.3675E+000

Figura 6.65: *Prueba t-Student AG_SGA vs. AG_FAMP en* f_{RR}.

6.6. Conclusiones

En este capítulo se ha presentado el primer escenario donde ha sido aplicado el sistema de optimización de parámetros propuesto: un AG con codificación binaria.

En primer lugar, se ha descrito la estructura del AG utilizado; un AG para la resolución de un problema de minimización sobre un conjunto de seis funciones representativas, utilizadas frecuentemente en la literatura como funciones de evaluación del comportamiento de un AG.

Seguidamente, se ha mostrado cómo se ha integrado el sistema de optimización propuesto con dicho AG binario.

Por último, se han presentado los resultados obtenidos. Para cada función, con el sistema propuesto, se han hallado los valores de los parámetros así como la forma en la que varían durante la ejecución del AG. Posteriormente, se ha comprobado el comportamiento del AG con dichos valores (a este algoritmo se le ha denominado AG_SGA) y se han comparado los resultados, mediante tests estadísticos, con los obtenidos al utilizar los parámetros estáticos recomendados en la literatura así como al utilizar otros AGs Adaptativos.

A la vista de los resultados obtenidos en dichos tests estadísticos, podemos concluir que AG_SGA ha sido el algoritmo más robusto, efectivo y versátil en la optimización de las seis funciones de prueba.

Capítulo 7

AG híbrido. Sistema borroso-genético

7.1. Introducción

En este capítulo se describe el segundo escenario en el que se ha aplicado el sistema de optimización propuesto, así como los experimentos realizados y los resultados obtenidos. En este caso se trata de un sistema borroso-genético, donde el propósito del sistema genético es hallar una base de conocimiento (BC) para que el sistema borroso que la incluya pueda resolver eficientemente el problema planteado. De nuevo, el objetivo del SGA es hallar un conjunto de parámetros que optimicen el comportamiento del AG.

Como se ha comentado previamente en el capítulo 4, a la hora de diseñar un sistema borroso-genético lo primero que se ha de decidir es qué parte de la base de conocimiento se desea que el AG optimice, ya que puede optimizarse la base de reglas, la base de datos o ambas. El espacio de búsqueda del AG difiere en función de lo que se quiera optimizar. A su vez, también es importante distinguir entre un proceso de aprendizaje y un proceso de ajuste. Un proceso de ajuste asume la existencia de una base de reglas predefinida y tiene como objetivo encontrar un conjunto de parámetros óptimos para las funciones de pertenencia y/o las funciones de escalado, es decir, los parámetros de la base de datos. Por otra parte, un proceso de aprendizaje constituye un método de diseño automático de una BC al completo o de algunos de sus componentes.

Por tanto, es posible obtener:

- sistemas con ajuste genético de la base de datos.
- sistemas con aprendizaje genético de la base de reglas.
- sistemas con aprendizaje genético de la BC.

En nuestro caso, el sistema borroso-genético es un sistema con aprendizaje genético de la BC, donde conjuntamente existe aprendizaje de las funciones de pertenencia y de la base de reglas. De los tres modelos existentes para el aprendizaje en sistemas borrosos basados en reglas se ha utilizado el modelo de Pittsburgh, donde la BC se codifica como un individuo y el AG aprende la mejor base asociada al problema a resolver.

7.2. Estructura del sistema borroso-genético

El sistema borroso-genético utilizado se corresponde con el esquema mostrado en la figura 7.1, formado por el controlador borroso, el modelo matemático del sistema a controlar, la población de bases de conocimiento, el sistema de evaluación y el sistema genético. A continuación se describen los distintos módulos del sistema.

Sistema a controlar

El sistema a controlar, en nuestro caso, es una referencia clásica en el campo de control de sistemas continuos, el péndulo invertido. Como se muestra en la figura 7.2, se trata de un vehículo que puede desplazarse horizontalmente dentro de unos límites y dispone, en su parte superior, de un péndulo colocado de forma invertida que puede desplazarse dentro de un margen angular desde la posición vertical. El objetivo consiste en posicionar al vehículo en la parte central y al péndulo en la vertical, partiendo de cualquier posición y ángulo de vehículo y péndulo respectivamente.

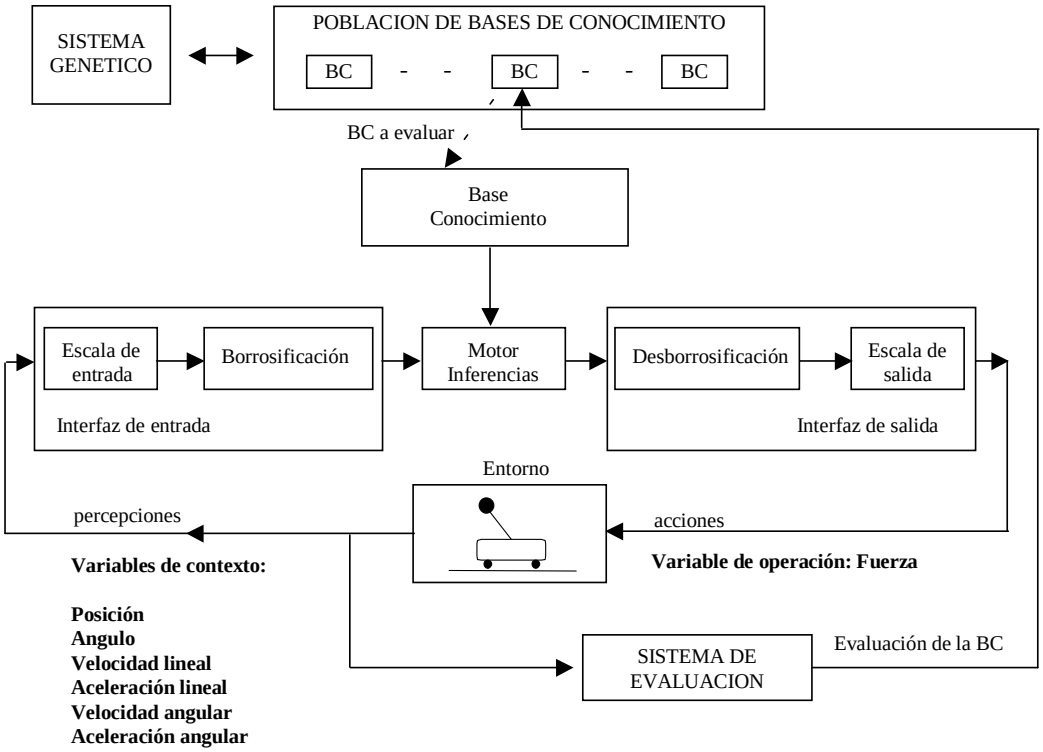

Figura 7.1: *Sistema borroso-genético utilizado.*

Para el péndulo invertido se han definido las siguientes variables de contexto:

- Posición, que representa la posición horizontal del vehículo.
- Velocidad lineal, que indica la velocidad del vehículo.
- Aceleración lineal. Aceleración del vehículo.
- Angulo, que representa la posición angular del péndulo respecto al eje vertical.
- Velocidad angular, que contiene la velocidad angular del péndulo.
- Aceleración angular, que indica la aceleración del péndulo.

Como variable de operación se ha definido la variable:

- Fuerza, que corresponde con la salida del sistema de control, y representa la acción que se realiza sobre el vehículo para lograr el objetivo.

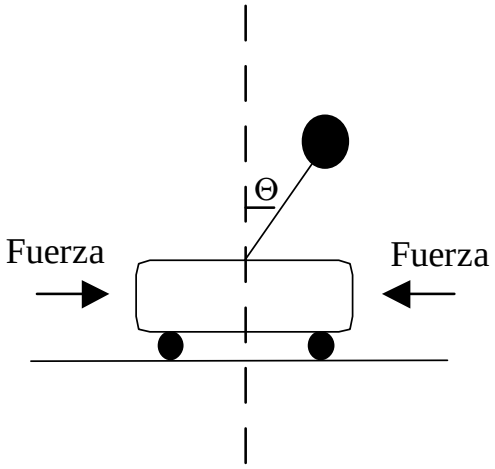

Figura 7.2: *Pendulo invertido.*

Controlador borroso

El controlador borroso tiene como finalidad realizar las acciones pertinentes, a partir del conocimiento que dispone, sobre el sistema a controlar para lograr el objetivo. Está formado por:

- Interfaz de entrada. Compuesto por una etapa de escalado y una etapa de borrosificación. El escalado a la entrada adapta el rango de los valores reales de las variables de contexto al universo del discurso de las variables lingüísticas. La borrosificación transforma los valores de las variables de contexto en información borrosa, permitiendo que el sistema pueda trabajar con datos reales.

- Base de conocimiento. Contiene el conocimiento sobre el control del péndulo invertido, estando formada por dos componentes, una base de datos y una base de reglas. La base de datos contiene información sobre las variables de contexto y operación del sistema, sobre las funciones de pertenencia asociadas a cada una de las variables y sobre las funciones de escalado. La base de reglas contiene el conjunto de reglas difusas de control del sistema.

- Motor de inferencias. Es el encargado de generar la acción que se efectuará sobre el sistema a controlar. Para generar la acción parte del estado del sistema (definido por el conjunto de variables de contexto) y de la BC. Como

resultado de aplicar el estado del sistema a la base de conocimiento, el motor de inferencias infiere los valores de la variable de operación, la fuerza.

- Interfaz de salida. Compuesto por una etapa de desborrosificación y una etapa de escalado a la salida. La desborrosificación adapta la salida del sistema borroso a valores numéricos normalizados de la variable de operación, la fuerza. La función de escalado transforma los valores normalizados al rango de la variable de operación.

Sistema de evaluación

El sistema de evaluación realiza una medida de la bondad del comportamiento de la BC en el sistema a controlar. El sistema de evaluación utilizado es el mostrado en [Cañada-Bago, 2004], que es un sistema de evaluación borroso basado en el modelo de Mamdani, en el que los antecedentes son los estados del sistema a controlar y el consecuente es la evaluación instantánea del estado del sistema.

Debido a que se trata de un sistema basado en conocimiento, el sistema de evaluación dispone de una BC de evaluación, que consta de la definición de las variables del sistema, definición de las funciones de pertenencia de las variables y de un conjunto de reglas de evaluación. En la determinación de las variables utilizadas en el sistema de evaluación se emplean únicamente las variables de contexto posición y ángulo, y una variable de salida para la evaluación. Por lo tanto, las evaluaciones instantáneas del sistema se realizarán a partir de estas dos variables, que corresponden con las utilizadas con el objetivo. Con objeto de simplificar el sistema de evaluación, el autor ha definido un número pequeño de conjuntos borrosos, en concreto tres, asociados a las variables posición y ángulo. Sin embargo, para poder disponer de un rango mayor en la variable de salida, la evaluación, ha definido nueve funciones de pertenencia para esta variable.

A partir del objetivo perseguido por el controlador borroso, las variables seleccionadas y sus conjuntos borrosos, el autor ha definido distintos estados del sistema a controlar, asociando una evaluación a cada estado, según se muestra en la figura 7.3.

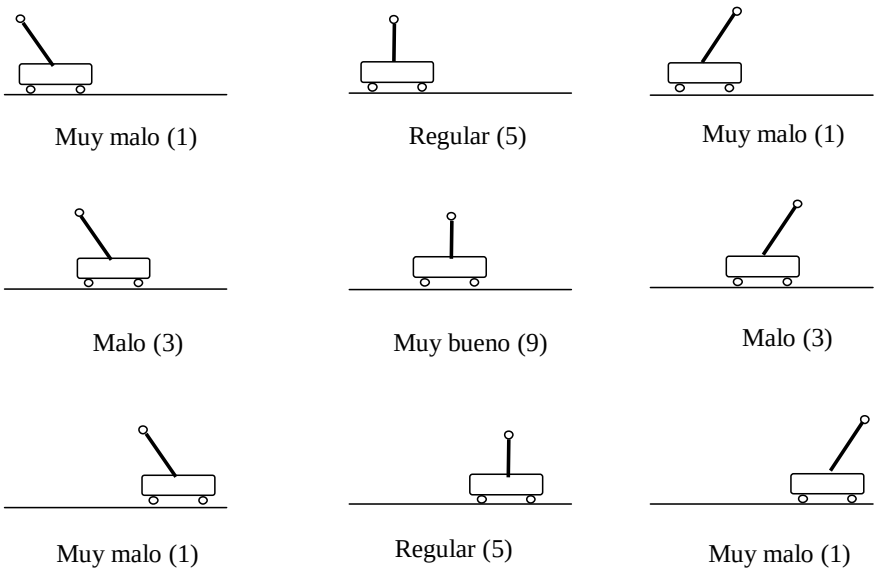

Figura 7.3: *Evaluaciones de los diferentes estados del sistema a controlar.*

El sistema borroso de evaluación permite, a partir del estado del sistema y de la BC de evaluación, inferir la evaluación instantánea. Para realizar la evaluación, se posiciona al vehículo y al péndulo en un estado inicial, permitiendo que el controlador actúe sobre el sistema. Como consecuencia, se obtiene un registro histórico de la evaluación como la evolución temporal de las evaluaciones instantáneas a partir del punto de partida.

Este registro histórico da una evaluación parcial de la BC, ya que se estudia la evolución del sistema desde un único punto de partida. Para evaluar la BC es necesario partir desde distintos puntos de partida. A partir de los registros de las evaluaciones instantáneas, realizados desde diferentes puntos de partida, se realiza la evaluación definitiva de la BC. En concreto, se ha realizado la evaluación desde un total de 16 puntos de partida diferentes, con todas las combinaciones posibles de las variables: posición, velocidad lineal, ángulo y velocidad angular en los valores normalizados 0.15 - 0.85, calculando la evaluación desde cada punto de partida como el valor medio de las evaluaciones instantáneas. Posteriormente, se ha calculado la evaluación media de las 16 evaluaciones de cada punto de partida, representando este valor el 80 % de la calificación definitiva de la BC. Si durante las 16 pruebas la BC ha consegue el objetivo, se le añade el 20 % restante en concepto de completitud.

Pto. Partida		
1	Valor medio de evaluación instantánea	
2	Valor medio de evaluación instantánea	
...	...	
...	...	
...	...	Valor
...	...	Medio
...	...	
...	...	
16	Valor medio de evaluación instantánea	

Evaluación final = 0.8 * Valor Medio + 0.2 * Completitud

Figura 7.4: *Cálculo de la evaluación final de la BC.*

Sistema genético

El sistema genético va a permitir obtener nuevas bases de conocimiento a partir de la población inicial. A la hora de implementar el AG se han definido los siguientes componentes:

1. Representación genética de las soluciones. La población del AG está formada por N individuos, siendo cada uno de ellos una BC. La codificación de cada BC se ha realizado según se describe en [Magdalena y Monasterio, 1995], distinguiendo entre la base de datos y la base de reglas. La base de datos se ha codificado mediante tres vectores de números reales: uno que codifica las dimensiones del controlador, otro que codifica los límites de normalización, y el tercero, que codifica las funciones de pertenencia. Para la codificación de la base de reglas se ha utilizado un vector de reglas, donde cada regla se representa mediante una cadena de bits (bits string), la cual se ha divido en dos partes: una para la codificación del antecedente y otra para la codificación del consecuente. En la figura 7.5 se muestra un ejemplo de una BC.

```
BASE ..\bases\bca09.bc FUERZA 0.0 VAR 7 GRUPOS 1 DEFECTO 0.500

VARIABLE POSICION CONJUNTOS 3
CONJUNTO 1 ( 0.000 0.000 ) ( 0.045 1.000 ) ( 0.469 1.000 ) ( 0.497 0.000 )
CONJUNTO 2 ( 0.336 0.000 ) ( 0.499 1.000 ) ( 0.517 1.000 ) ( 0.575 0.000 )
CONJUNTO 3 ( 0.517 0.000 ) ( 0.575 1.000 ) ( 0.999 1.000 ) ( 1.000 0.000 )

VARIABLE ANGULO CONJUNTOS 3
CONJUNTO 1 ( 0.000 0.000 ) ( 0.001 1.000 ) ( 0.350 1.000 ) ( 0.500 0.000 )
CONJUNTO 2 ( 0.393 0.000 ) ( 0.458 1.000 ) ( 0.571 1.000 ) ( 0.906 0.000 )
CONJUNTO 3 ( 0.648 0.000 ) ( 0.919 1.000 ) ( 0.950 1.000 ) ( 1.000 0.000 )

VARIABLE VELOC_ANG CONJUNTOS 3
CONJUNTO 1 ( 0.000 0.000 ) ( 0.125 1.000 ) ( 0.435 1.000 ) ( 0.440 0.000 )
CONJUNTO 2 ( 0.131 0.000 ) ( 0.499 1.000 ) ( 0.501 1.000 ) ( 0.650 0.000 )
CONJUNTO 3 ( 0.501 0.000 ) ( 0.650 1.000 ) ( 0.999 1.000 ) ( 1.000 0.000 )

VARIABLE ACEL_ANG CONJUNTOS 3
CONJUNTO 1 ( 0.000 0.000 ) ( 0.144 1.000 ) ( 0.199 1.000 ) ( 0.381 0.000 )
CONJUNTO 2 ( 0.355 0.000 ) ( 0.444 1.000 ) ( 0.475 1.000 ) ( 0.768 0.000 )
CONJUNTO 3 ( 0.548 0.000 ) ( 0.780 1.000 ) ( 0.801 1.000 ) ( 1.000 0.000 )

VARIABLE ACEL_LIN CONJUNTOS 3
CONJUNTO 1 ( 0.000 0.000 ) ( 0.001 1.000 ) ( 0.350 1.000 ) ( 0.500 0.000 )
CONJUNTO 2 ( 0.371 0.000 ) ( 0.475 1.000 ) ( 0.503 1.000 ) ( 0.775 0.000 )
CONJUNTO 3 ( 0.503 0.000 ) ( 0.863 1.000 ) ( 0.932 1.000 ) ( 1.000 0.000 )

VARIABLE VELOC_LIN CONJUNTOS 3
CONJUNTO 1 ( 0.000 0.000 ) ( 0.159 1.000 ) ( 0.243 1.000 ) ( 0.393 0.000 )
CONJUNTO 2 ( 0.371 0.000 ) ( 0.475 1.000 ) ( 0.503 1.000 ) ( 0.775 0.000 )
CONJUNTO 3 ( 0.503 0.000 ) ( 0.863 1.000 ) ( 0.932 1.000 ) ( 1.000 0.000 )

VARIABLE FUERZA CONJUNTOS 9
CONJUNTO 1 ( 0.000 0.000 ) ( 0.004 1.000 ) ( 0.047 1.000 ) ( 0.123 0.000 )
CONJUNTO 2 ( 0.085 0.000 ) ( 0.125 1.000 ) ( 0.234 1.000 ) ( 0.238 0.000 )
CONJUNTO 3 ( 0.126 0.000 ) ( 0.250 1.000 ) ( 0.351 1.000 ) ( 0.375 0.000 )
CONJUNTO 4 ( 0.351 0.000 ) ( 0.375 1.000 ) ( 0.376 1.000 ) ( 0.500 0.000 )
CONJUNTO 5 ( 0.499 0.000 ) ( 0.501 1.000 ) ( 0.519 1.000 ) ( 0.531 0.000 )
CONJUNTO 6 ( 0.523 0.000 ) ( 0.624 1.000 ) ( 0.626 1.000 ) ( 0.750 0.000 )
CONJUNTO 7 ( 0.684 0.000 ) ( 0.766 1.000 ) ( 0.860 1.000 ) ( 0.866 0.000 )
CONJUNTO 8 ( 0.768 0.000 ) ( 0.874 1.000 ) ( 0.876 1.000 ) ( 0.900 0.000 )
CONJUNTO 9 ( 0.879 0.000 ) ( 0.957 1.000 ) ( 0.965 1.000 ) ( 1.000 0.000 )

GRUPO FUERZA REGLAS 15
REGLA R1 ( ACEL_ANG 3 ) ( ACEL_LIN 2 ) ( FUERZA 8 )
REGLA R2 ( ANGULO 2 ) ( ACEL_LIN 1 ) ( FUERZA 2 )
REGLA R3 ( ANGULO 1 ) ( FUERZA 2 )
REGLA R4 ( ANGULO 1 ) ( VELOC_ANG 1 ) ( FUERZA 7 )
REGLA R5 ( ACEL_ANG 1 ) ( VELOC_ANG 3 ) ( FUERZA 3 )
REGLA R6 ( VELOC_LIN 1 ) ( ANGULO 1 ) ( FUERZA 7 )
REGLA R7 ( ANGULO 1 ) ( POSICION 2 ) ( FUERZA 1 )
REGLA R8 ( ANGULO 1 ) ( VELOC_ANG 2 ) ( FUERZA 2 )
REGLA R9 ( ACEL_ANG 1 ) ( VELOC_ANG 3 ) ( FUERZA 5 )
REGLA R10 ( POSICION 2 ) ( FUERZA 8 )
REGLA R11 ( ANGULO 1 ) ( POSICION 2 ) ( FUERZA 2 )
REGLA R12 ( VELOC_LIN 3 ) ( ANGULO 1 ) ( FUERZA 7 )
REGLA R13 ( POSICION 3 ) ( FUERZA 7 )
REGLA R14 ( ANGULO 3 ) ( VELOC_ANG 2 ) ( FUERZA 5 )
REGLA R15 ( ACEL_LIN 1 ) ( VELOC_ANG 1 ) ( FUERZA 7 )
```

Figura 7.5: *Ejemplo de una base de conocimiento.*

2. Método de formación de la población inicial. Los individuos se generan a partir de conocimiento humano, incorporándose éste a través de las reglas de actuación. La población inicial está compuesta por bases de conocimiento idénticas con conocimiento nulo, es decir, la evaluación inicial de todos los individuos es 0.0.

3. Sistema de evaluación. Este evalúa el grado de adaptación de las bases de conocimiento al entorno. Se corresponde con el descrito anteriormente.

4. Método de selección.

   Para la selección de los individuos que actúan como padres se han empleado los siguientes métodos: *Proporcional Selection* -selección proporcional-, para el cálculo de las probabilidades de reproducción y, *Stochastic Sampling with Replacement* -ruleta-, como algoritmo de muestreo. De esta forma tienen mayor probabilidad de ser seleccionados aquellos individuos que poseen una mayor evaluación. El porcentaje de individuos seleccionados viene dado por la P_s.

5. Operadores genéticos. Durante la fase de reproducción se han utilizado los siguientes operadores:

   - Cruce.

     El cruce de los individuos seleccionados se realiza mediante el intercambio de las reglas difusas que contienen. El proceso de selección de un grupo de reglas en un individuo lleva implícito la inclusión de las variables utilizadas en las mismas así como los conjuntos borrosos asociados a cada variable. El operador de cruce opera de forma similar al operador *Single Point Crossover* -cruce de 1 punto-. El proceso se realiza de la siguiente forma:

     a) se escogen las bases de reglas, r_i y r_j, de 2 bases de conocimiento, BC_i y BC_j, obtenidas mediante el método de selección:

     $$r_i = r_{i1}, \ldots, r_{ik}$$

     $$r_j = r_{j1}, \ldots, r_{il}$$

     b) se eligen 2 números aleatorios, α y β, que se utilizan como puntos de cruce:

     $$r_i = r_{i1}, \ldots, r_{i\alpha}, |r_{i\alpha+1}, \ldots\ldots, r_{ik}$$

     $$r_j = r_{j1}, \ldots\ldots, r_{j\beta}, |r_{j\beta+1}, \ldots, r_{jl}$$

c) se realiza el cruce, teniendo como nuevas bases de reglas:

$$r_u = r_{i1}, \ldots, r_{i\alpha}, |r_{i\beta+1}, \ldots, r_{jl}$$

$$r_v = r_{j1}, \ldots\ldots, r_{j\beta}, |r_{j\alpha+1}, \ldots\ldots, r_{ik}$$

El número de cruces que se realizan viene dado por la Probabilidad de Cruce, P_c.

- Mutación. Con este operador se alteran aleatoriamente los componentes de los nuevos individuos obtenidos tras el proceso de cruce. Existen tres tipos de mutaciones:

  a) *Mutación de puntos* que definen los conjuntos borrosos. Los conjuntos borrosos utilizados son trapecios definidos por cuatro puntos. El procedimiento para realizar la mutación es el siguiente: dada una BC elegida al azar, se selecciona, también al azar, un conjunto borroso de cualquiera de las variables que están definidas en dicha base. La variación de la posición de cada punto se hace en un intervalo determinado por los dos puntos adyacentes. Como puede apreciarse en la figura 7.6, el punto *b2* puede desplazarse en el intervalo que comienza en el punto *d1* y acaba en el punto *c2*. El número de mutaciones viene dado por la Probabilidad de Mutación de Puntos, P_{mp}.

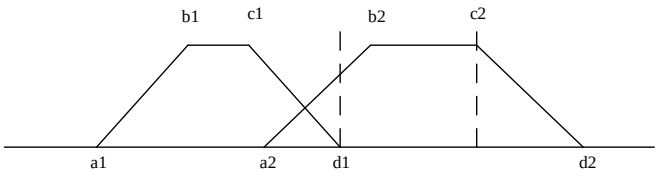

Figura 7.6: *Mutación de puntos en un conjunto borroso.*

  b) *Mutación de conjuntos borrosos* en las reglas. Mediante este operador se mutan los conjuntos borrosos utilizados en los antecedentes y consecuentes de las reglas de actuación. Así, se elige, al azar, un conjunto borroso de una variable de una regla de actuación de una BC y se cambia por otro conjunto borroso de la misma variable. El número de mutaciones viene dado por la Probabilidad de Mutación de Conjuntos Borrosos, P_{mcb}.

Antes de la mutación
REGLA R1 (ANGULO 1)(POSICION 3)(FUERZA 6)

Después de la mutación
REGLA R1 (ANGULO 3)(POSICION 3)(FUERZA 6)

c) *Mutación de variables* en las reglas. Este operador altera las variables utilizadas en los antecedentes de las reglas de actuación, permitiendo que la nueva BC tenga variables diferentes a las de sus antecesores. El proceso se realiza de la siguiente forma: se elige una variable del antecedente de una regla de actuación de una BC y se cambia por otra variable del antecedente. El número de mutaciones viene dado por la Probabilidad de Mutación de Variables, P_{mv}.

Antes de la mutación
REGLA R1 (ANGULO 1)(POSICION 3)(FUERZA 6)

Después de la mutación
REGLA R1 (VELOC_LIN 1)(POSICION 3)(FUERZA 6)

6. Valores de los parámetros.

   El tamaño de la población es de 30 individuos. Se ha elegido un valor pequeño debido al elevado coste computacional que conlleva la evalución cada BC. El resto de valores de los parámetros se corresponden con los de cada individuo CP_i del SGA. Inicialmente son valores aleatorios (a excepción de CP1) y evolucionan hacia valores más óptimos al aplicar el SGA propuesto.

7. Método de sustitución. La sustitución de los individuos de la población por los nuevos individuos generados se ha realizado mediante el algoritmo de la ruleta inversa, por el cual, la probabilidad de ser eliminados es inversamente proporcional a su bondad y, por tanto, los individuos con mayor evaluación tienen menor posibilidad de ser eliminados. También se ha utilizado una estrategia elitista, donde no se reemplaza a la mejor BC de la población. Además, la sustitución se lleva a cabo si la evaluación de los nuevos individuos generados es superior a la de los que sustituyen.

7.3. Influencia de la población inicial

En este contexto, y previamente a integrar el Sistema Genético Adicional con el sistema borroso-genético, se han realizado una serie de experimentos con un doble objetivo: por un lado, conocer si la población inicial influye en el comportamiento del AG; y por otro, comprobar si el rendimiento el AG es satisfactorio, con independencia de la población inicial de partida, al utilizar los parámetros recomendados en la literatura. Para ello, se ha ejecutado el AG con diferentes poblaciones iniciales, donde las modificaciones realizadas han afectado:

- al contenido de la base de datos (manteniendo la misma base de reglas).

  En este caso, se ha mantenido constante el tamaño de la población, N=30, y se ha modificado el número de conjuntos borrosos de las variables de contexto de las bases de datos. Así, se han realizado 4 experimentos con 4 poblaciones iniciales. Cada población está compuesta por 30 bases de conocimiento idénticas, con conocimiento nulo (evaluación inicial 0.0). En cada una de estas poblaciones se ha modificado el número de conjuntos borrosos de las variables de contexto de cada base de datos, siendo 3, 5, 7 y 9 los conjuntos borrosos iniciales. Estos 4 experimentos se denotan como E_1, E_2, E_3 y E_4 respectivamente.

- al tamaño de la población (manteniendo constante el número de conjuntos borrosos).

  Ahora, las diferentes poblaciones iniciales que han sido utilizadas son las siguientes:

  1. Población compuesta por N=12 bases de conocimiento idénticas, con conocimiento nulo y con 3 conjuntos borrosos para cada variable de contexto. Experimento E_5.
  2. Población compuesta por N=30 bases de conocimiento idénticas, con conocimiento nulo y con 3 conjuntos borrosos para cada variable de contexto. Experimento $E_6 = E_1$.
  3. Población compuesta por N=50 bases de conocimiento idénticas, con conocimiento nulo y con 3 conjuntos borrosos para cada variable de contexto. Experimento E_7.

4. Población compuesta por N=100 bases de conocimiento idénticas, con conocimiento nulo y con 3 conjuntos borrosos para cada variable de contexto. Experimento E_8.

Para cada una de las 8 poblaciones iniciales se ha ejecutado el AG 30 veces con un máximo de 5000 evaluaciones. Los valores de los parámetros utilizados en cada caso han sido los siguientes:

a) Para una población de N=12 y N=30:

- P_s=1.0
- P_c=0.95
- P_{mp}=0.01
- P_{mcb}=0.01
- P_{mv}=0.01

b) Para N=50:

- P_s=1.0
- P_c=0.6
- P_{mp}=0.001
- P_{mcb}=0.001
- P_{mv}=0.001

c) Para N=100:

- P_s=1.0
- P_c=0.65
- P_{mp}=0.008
- P_{mcb}=0.008
- P_{mv}=0.008

Al finalizar cada ejecución en cada experimento se ha obtenido una población diferente a la inicial y se ha seleccionado la BC que ha obtenido la mayor evaluación.

Resultados

En las tablas 7.1 y 7.2 se presentan los resultados obtenidos. En la tabla 7.1 se muestra la evaluación de la mejor BC obtenida al finalizar la ejecución de cada AG en los experimentos E1, E2, E3 y E4, donde se ha variado el contenido de la base de datos. Puede observarse cómo a medida que se ha incrementado el número de conjuntos borrosos en las bases de conocimiento iniciales, la evaluación (en media) de la mejor BC ha sido menor. Esto es debido a que la base de reglas inicial ha sido diseñada para tres conjuntos borrosos. Por tanto, para obtener un mejor comportamiento del AG, parece lógico incrementar la probabilidad de mutación a medida que existan más conjuntos borrosos en la base de datos .

Por otro lado, en la tabla 7.2 se muestra la evaluación de la mejor BC obtenida al finalizar cada AG en los experimentos E5, E6, E7 y E8, donde se ha variado el número de individuos en la población. A medida que se ha incrementado el número de bases de conocimiento se ha obtenido un mejor comportamiento en el AG. Para N=50 y N=100, la evaluación (en media) de la mejor BC es prácticamente la misma; sin embargo, en el segundo caso el aprendizaje se ha realizado de forma más rápida, ya que se ha obtenido el mismo resultado con un menor número de evaluaciones.

Por tanto, a la vista de los resultados, se puede afirmar que los valores de los parámetros utilizados en el AG no han sido los adecuados en todos los experimentos, fundamentalmente en E2, E3, E4 y E5. En estos casos se debería haber utilizado otros parámetros diferentes a los recomendados, por lo que, también en este escenario, es necesario un sistema que optimice dichos valores.

Tabla 7.1: *Evaluaciones de las bases de conocimiento al variar la base de datos.*

Nº ejecucion AG	**Exp. E1**	**Exp. E2**	**Exp. E3**	**Exp. E4**
1	0.758	0.280	0.663	0.665
2	0.255	0.690	0.710	0.363
3	0.706	0.710	0.269	0.712
4	0.665	0.422	0.723	0.327
5	0.713	0.681	0.700	0.397
6	0.732	0.722	0.319	0.274
7	0.665	0.293	0.390	0.672
8	0.706	0.398	0.721	0.700
9	0.692	0.682	0.357	0.700
10	0.666	0.683	0.357	0.249
11	0.733	0.671	0.709	0.384
12	0.723	0.293	0.698	0.687
13	0.666	0.374	0.258	0.285
14	0.726	0.688	0.726	0.399
15	0.712	0.698	0.391	0.274
16	0.725	0.742	0.384	0.379
17	0.665	0.663	0.392	0.353
18	0.726	0.668	0.672	0.706
19	0.665	0.706	0.746	0.729
20	0.753	0.293	0.484	0.402
21	0.719	0.711	0.392	0.343
22	0.731	0.670	0.190	0.283
23	0.740	0.277	0.717	0.729
24	0.745	0.681	0.720	0.724
25	0.758	0.736	0.461	0.684
26	0.742	0.692	0.678	0.289
27	0.754	0.716	0.684	0.270
28	0.694	0.279	0.721	0.688
29	0.745	0.688	0.419	0.297
30	0.744	0.668	0.731	0.726
MEDIA	**0.700**	**0.582**	**0.546**	**0.489**

Tabla 7.2: *Evaluaciones de las bases de conocimiento al variar el nº de individuos.*

Nº ejecucion AG	**Exp. E5**	**Exp. E6**	**Exp. E7**	**Exp. E8**
1	0.665	0.758	0.72	0.721
2	0.708	0.255	0.761	0.744
3	0.734	0.706	0.722	0.733
4	0.665	0.665	0.753	0.713
5	0.658	0.713	0.672	0.719
6	0.665	0.732	0.666	0.720
7	0.665	0.665	0.706	0.717
8	0.768	0.706	0.722	0.742
9	0.729	0.692	0.715	0.677
10	0.712	0.666	0.715	0.711
11	0.755	0.733	0.738	0.718
12	0.665	0.723	0.731	0.723
13	0.390	0.666	0.724	0.734
14	0.665	0.726	0.739	0.733
15	0.666	0.712	0.711	0.722
16	0.665	0.725	0.706	0.734
17	0.666	0.665	0.730	0.717
18	0.717	0.726	0.716	0.723
19	0.699	0.665	0.666	0.665
20	0.665	0.753	0.722	0.740
21	0.665	0.719	0.665	0.720
22	0.666	0.731	0.732	0.716
23	0.665	0.740	0.735	0.720
24	0.666	0.745	0.732	0.747
25	0.665	0.758	0.730	0.746
26	0.665	0.742	0.746	0.722
27	0.666	0.754	0.709	0.665
28	0.753	0.694	0.726	0.749
29	0.666	0.745	0.713	0.731
30	0.666	0.744	0.733	0.728
MEDIA	**0.675**	**0.700**	**0.718**	**0.721**

7.4. Integración con el sistema de optimización

Para hallar los parámetros que optimizan el comportamiento del AG en el sistema borroso-genético se ha integrado dicho sistema con el SGA, como se muestra en la figura 7.7.

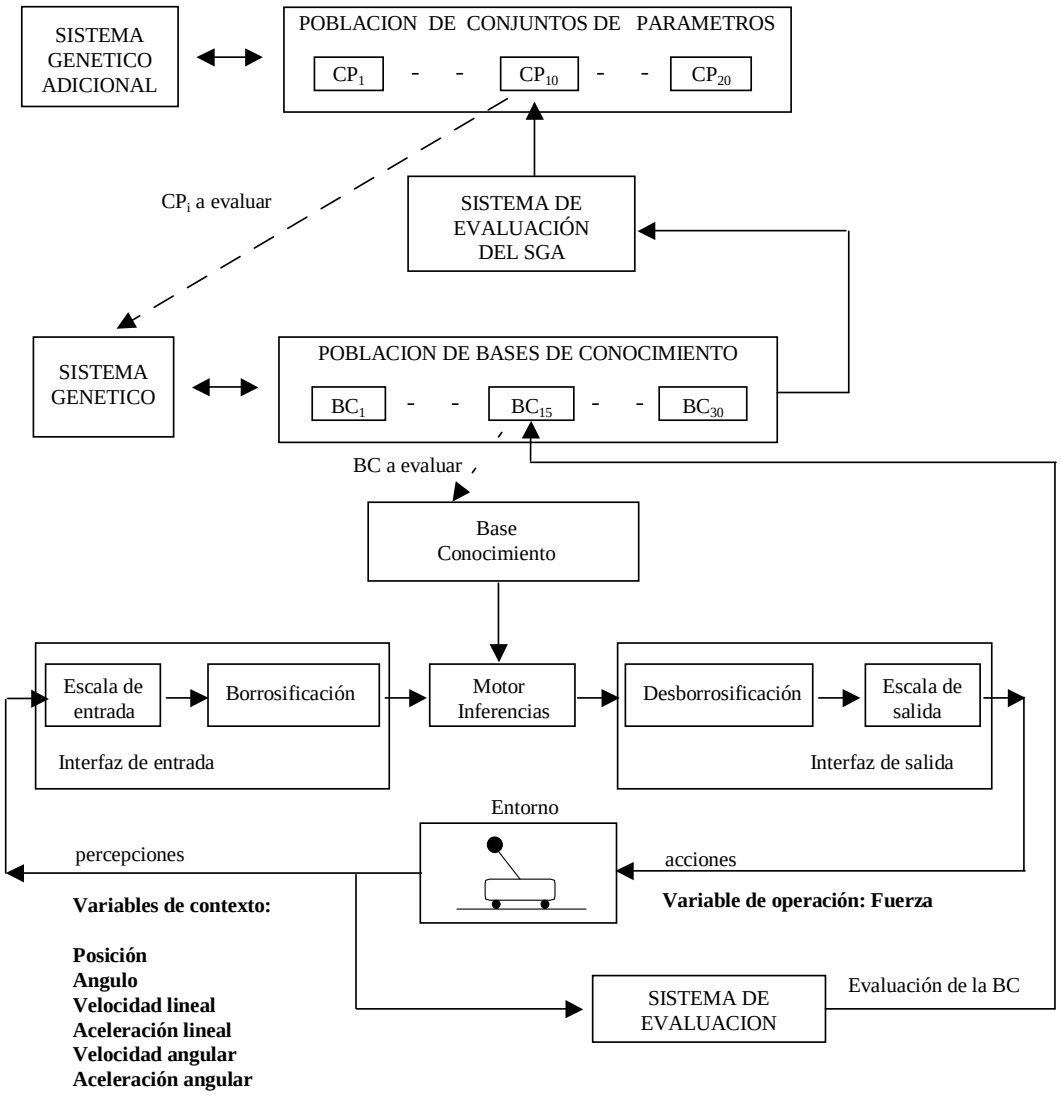

Figura 7.7: *Aprendizaje evolutivo del Conjunto de Parámetros en un sistema borroso-genético.*

En este caso, los parámetros que se desean optimizar son: P_s, P_c, P_{mp}, P_{mcb} y P_{mv}, por lo que la representación de cada CP_i se corresponde con un vector como el siguiente:

$$CP_i = \{P_{is}, P_{ic}, P_{imp}, P_{imcb}, P_{imv}, a_{i1}, b_{i1}, a_{i2}, b_{i2}, a_{i3}, b_{i3}, a_{i4}, b_{i4}, a_{i5}, b_{i5}\}$$

Las variaciones de los parámetros durante la ejecución del AG vienen dadas, en cada CP_i, por las funciones:

$$P_{is}(t) = p_0^{is}(1 - \frac{(Ln(t+1))^{(1/a_{i1})}}{(b_{i1}Ln(T+1))^{(1/a_{i1})}}) \quad \begin{matrix} a_{i1} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{ic}(t) = p_0^{ic}(1 - \frac{(Ln(t+1))^{(1/a_{i2})}}{(b_{i2}Ln(T+1))^{(1/a_{i2})}}) \quad \begin{matrix} a_{i2} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{imp}(t) = p_0^{imp}(1 - \frac{(Ln(t+1))^{(1/a_{i3})}}{(b_{i3}Ln(T+1))^{(1/a_{i3})}}) \quad \begin{matrix} a_{i3} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{imcb}(t) = p_0^{imcb}(1 - \frac{(Ln(t+1))^{(1/a_{i4})}}{(b_{i4}Ln(T+1))^{(1/a_{i3})}}) \quad \begin{matrix} a_{i3} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{imv}(t) = p_0^{imv}(1 - \frac{(Ln(t+1))^{(1/a_{i5})}}{(b_{i5}Ln(T+1))^{(1/a_{i3})}}) \quad \begin{matrix} a_{i3} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

Por tanto, el proceso de aprendizaje de los parámetros del AG así como de los parámetros $a_{i\lambda}$ y $b_{i\lambda}$ de cada función $p_{i\lambda}(t)$ se realiza de la siguiente forma:

En primer lugar, se parte de una población inicial de Conjuntos de Parámetros, CP(0), compuesta por 20 individuos CP_i.

Posteriormente, mientras que el número de iteraciones en el SGA (it_sga) sea inferior a 50:

- se evalúa la población CP(it_sga).

  Para evaluar cada CP_i de la población se ejecuta un AG, que utiliza los valores de los parámetros que contiene dicho CP_i. El AG parte de una población inicial P(0) de bases de conocimiento y tras realizarse 50 iteraciones se obtiene una nueva población, P(50). A partir de los individuos obtenidos en P(50) se evalúa CP_i.

- Una vez evaluados todos los individuos de CP(it_sga), se incrementa el número de iteraciones en el SGA (it_sga=it_sga+1).
- Seguidamente, se selecciona una nueva población CP(it_sga) a partir de CP (it_sga-1).
- A continuación, se aplican los operadores de cruce y mutación sobre CP(it_sga).
- Por último, se sustituye parte de CP(it_sga-1) por la descendencia generada.

Al final de la ejecución del Sistema Genético Adicional se obtiene una población diferente a la inicial, CP(50), y el CP_i cuya evaluación es la mayor, es el que contiene los mejores valores.

7.5. Resultados

En esta sección se presentan los resultados obtenidos en este segundo escenario donde ha sido aplicado el sistema de optimización propuesto. Para la optimización de los parámetros del AG, así como de los parámetros a_i y b_i de cada una de las funciones $p_i(t)$, se ha añadido el Sistema Genético Adicional al sistema de aprendizaje evolutivo de una base de conocimiento basado en el enfoque de Pittsburgh. En este caso, se han utilizado cinco funciones $p_i(t)$ i=1,...,5 que han afectado, de forma independiente, a los parámetros P_s, P_c, P_{mp}, P_{mcb} y P_{mv} respectivamente. Se han ejecutado 50 iteraciones, tanto en el AG del Sistema Genético como en el Sistema Genético Adicional, y se ha seleccionado el CP_i que ha obtenido la mayor evaluación al final del experimento.

7.5.1. Parámetros hallados con el sistema propuesto

Los valores del mejor CP_i obtenido se muestran en la tabla 7.3. Como puede apreciarse, el sistema ha encontrado como mejor AG a un modelo estacionario y adaptativo, donde el valor inicial de P_s es de 0.65.

Tabla 7.3: *Valores de los parámetros hallados en el sistema borroso-genético.*

Parámetro	**Valor**
P_s	0.65
P_c	0.375
P_{mp}	0.100
P_{mcb}	0.032
P_{mv}	0.040
a_1 - b_1	0.62 - 8.50
a_2 - b_2	1.17 - 7.25
a_3 - b_3	1.59 - 15.75
a_4 - b_4	0.80 - 4.00
a_5 - b_5	1.45 - 3.75

La variación de P_s y P_c durante la ejecución de 500 iteraciones en el AG puede observarse en la figura 7.8, mientras que la de P_{mp}, P_{mcb} y P_{mv} se muestra en la figura 7.9. Los parámetros que sufren una mayor atenuación, con respecto a su valor inicial, son P_{mv} y P_{mcb}, siendo esta del 27.5 % y del 25 % respectivamente.

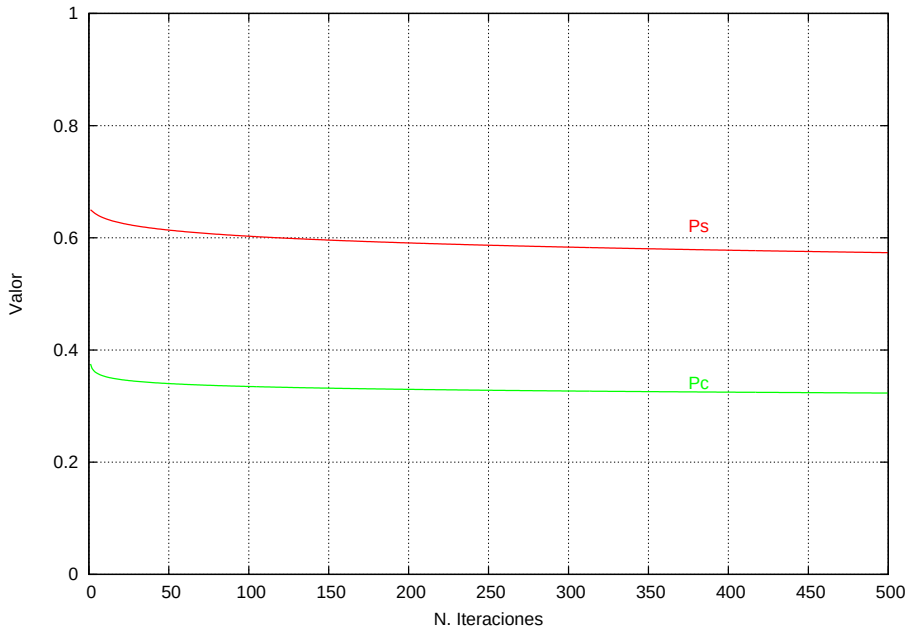

Figura 7.8: *Evolución de P_s y P_c durante la ejecución del AG en un sistema borroso-genético.*

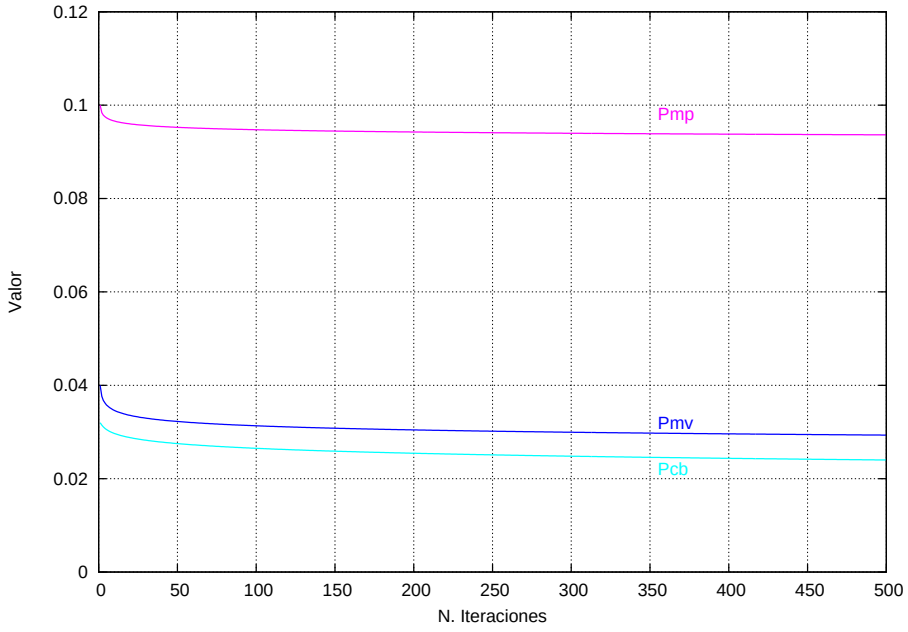

Figura 7.9: *Evolución de P_{mp}, P_{mcb} y P_{mv} durante la ejecución del AG en un sistema borroso-genético.*

El parámetro que menor atenuación tiene es P_{mp}, con una disminución del 6.4 %, al que le siguen P_s, con el 11.8 %, y P_c, con el 13.8 %. El hecho de que los parámetros P_{mv} y P_{mcb} sufran la mayor atenuación puede ser debido a que estos sean los más influyentes en el comportamiento del AG. Además, también es de destacar el que todas las probabilidades de mutación tengan un valor mayor a los recomendados. En especial, llama la atención el valor de P_{mp}; 0.1 al inicio de la ejecución y 0.093 al final, lo cual puede deberse a dos motivos:

1) al operador de mutación de puntos utilizado, ya que la mutación de los puntos que definen los conjuntos borrosos se ha realizado variando la posición de cada punto en un intervalo determinado por los dos puntos adyacentes, por lo que ello puede haber mitigado el efecto de una probabilidad tan alta.

2) a que este parámetro sea el menos influyente en el comportamiento del AG.

Por otro lado, al igual que en el escenario anterior, la mayor atenuación de las probabilidades de mutación se produce en la etapa inicial de la ejecución del AG. En las primeras 50 iteraciones, P_{mp}, P_{mcb} y P_{mv} se han atenuado, con respecto a su valor inicial, el 5 %, 15.6 % y 20 % respectivamente, por lo que la mayor parte de la exploración de la información se realiza en dicha etapa.

7.5.2. Comportamiento del AG con los parámetros hallados

A continuación, se han llevado a cabo 30 ejecuciones del AG_SGA, con un máximo de 500 iteraciones cada una, con el fin de evaluar su comportamiento en el sistema borroso-genético. En la tabla 7.4 se muestran los resultados obtenidos. La primera columna identifica el experimento realizado; la segunda, la evaluación de la mejor BC obtenida al final de cada ejecución del AG_SGA; y la tercera, el número de iteraciones a partir de la cual no existe una mejora en la solución encontrada. Cuanto mayor es la evaluación de la BC obtenida, mejor es su comportamiento en el control del sistema. Como puede observarse, la mejor BC tiene una evaluación de 0.808 y la peor 0.728. A priori, la evaluación media obtenida parece ser elevada; sin embargo, para conocer realmente la bondad del AG_AGS es necesario comparar su comportamiento con otros AGs.

Tabla 7.4: *Evaluación de las bases de conocimiento en el sistema borroso-genético.*

Nº Exp.	**Evaluación**	**Nº iteración**
1	0.775	492
2	0.779	491
3	0.776	453
4	0.782	289
5	0.801	380
6	0.765	494
7	0.728	443
8	0.78	401
9	0.759	338
10	0.764	490
11	0.772	436
12	0.756	448
13	0.755	490
14	0.798	300
15	0.790	310
16	0.785	405
17	0.770	464
18	0.808	443
19	0.791	482
20	0.788	422
21	0.753	443
22	0.796	480
23	0.790	489
24	0.790	422
25	0.765	497
26	0.755	499
27	0.770	459
28	0.800	421
29	0.801	454
30	0.792	494
MEDIA	**0.777**	**438**

7.5.3. Comportamiento de otros AGs Adaptativos

Los principales métodos de adaptación de la probabilidad de mutación han sido adaptados para que P_{mp}, P_{mcb} y P_{mv} varíen en el intervalo [0.005, 0.01], que es el propuesto en [Schaffer et al., 1989] para una población comprendida entre 20 y 30 individuos. Todos los algoritmos se han ejecutado 30 veces, con un máximo de 500 iteraciones, al igual que el AG_SGA. Los mecanismos adaptativos utilizados han sido:

- Control determinístico (AG_DET), donde P_{mp}, P_{mcb} y P_{mv} han variado su valor de la siguiente forma:

$$P_{mp}(t) = P^h_{mp} - \frac{P^h_{mp}-P^l_{mp}}{T}t \quad 0 \leq \mathrm{t} \leq \mathrm{T}$$

$$P_{mcb}(t) = P^h_{mcb} - \frac{P^h_{mcb}-P^l_{mcb}}{T}t \quad 0 \leq \mathrm{t} \leq \mathrm{T}$$

$$P_{mv}(t) = P^h_{mv} - \frac{P^h_{mv}-P^l_{mv}}{T}t \quad 0 \leq \mathrm{t} \leq \mathrm{T}$$

  siendo P^h_{mp}=0.01 , P^l_{mp}=0.005, P^h_{mcb}=0.01, P^l_{mcb}=0.005, P^h_{mv}=0.01 , P^l_{mv}=0.005 y T=500.

- Control adaptativo a nivel del individuo de P_{mp}, P_{mcb} y P_{mv} (AG_IL), en donde cada BC de la población ha tenido asociada unas P_{mp}, P_{mcb} y P_{mv} que han variado de la siguiente forma:

$$P_{mp} = \begin{cases} 0.01 & \text{si } f' < \overline{f} \\ 0.01 * \frac{f_{max}-f'}{f_{max}-\overline{f}} & \text{si } \overline{f} \leq f' < f_{max} \\ 0.005 & \text{si } f' = f_{max} \end{cases}$$

$$P_{mcb} = \begin{cases} 0.01 & \text{si } f' < \overline{f} \\ 0.01 * \frac{f_{max}-f'}{f_{max}-\overline{f}} & \text{si } \overline{f} \leq f' < f_{max} \\ 0.005 & \text{si } f' = f_{max} \end{cases}$$

$$P_{mv} = \begin{cases} 0.01 & \text{si } f' < \overline{f} \\ 0.01 * \frac{f_{max}-f'}{f_{max}-\overline{f}} & \text{si } \overline{f} \leq f' < f_{max} \\ 0.005 & \text{si } f' = f_{max} \end{cases}$$

donde f' es la evaluación de la BC, f_{max} es la evaluación de la mejor BC en la población y $\overline{f}$ es la evaluación media de la población.

- Control adaptativo a nivel de la población de P_{mp}, P_{mcb} y P_{mv} (AG_FAMP) mediante la utilización un controlador borroso que ha adaptado las probabilidades durante la ejecución del AG. El controlador borroso se ha disparado cada G=10 iteraciones y las probabilidades de mutación han permanecido fijas durante ese tiempo.

- Control auto-adaptativo a nivel del individuo de P_{mp}, P_{mcb} y P_{mv} (AG_SELF), donde las probabilidades de mutación han sido codificadas junto a las BC de la población y han evolucionado junto a ellas. Para la configuración del algoritmo se han elegido los valores: δ=0.001, P^l_{mp}=0.005, P^h_{mp}=0.01, P^l_{mcb}=0.005, P^h_{mcb}=0.01, P^l_{mv}=0.005 y P^h_{mv}=0.01.

- Control Aleatorio (AG_RAN), donde en cada iteración del AG las probabilidades de mutación han sido elegidas aleatoriamente en el intervalo [0.005, 0.01].

Al igual que en el escenario anterior, también se ha comprobado el comportamiento del AG al utilizar probabilidades de mutación estáticas. Se han utilizado tres probabilidades distintas, pertenecientes al intervalo [0.005, 0.01]:

1. P_{mp}=0.005, P_{mcb}=0.005, P_{mv}=0.005 (AG_EST1)
2. P_{mp}=0.075, P_{mcb}=0.0075, P_{mv}=0.0075 (AG_EST2)
3. P_{mp}=0.01, P_{mcb}=0.01, P_{mv}=0.01 (AG_EST3)

Recordar que para una población de 30 individuos, como es nuestro caso, el valor recomendado de P_m es 0.01. En todos los casos, la probabilidad de selección y la probabilidad de cruce han sido las recomendadas en [Schaffer et al., 1989]: P_s=1.0 y P_c=0.95. La tabla 7.5 muestra el resumen de los resultados obtenidos en cada algoritmo, donde la primera columna identifica el algoritmo utilizado; la segunda, la media de la mejor BC obtenida al final de cada ejecución; y la tercera, la media del número de iteraciones a partir de la cual no ha existido una mejora en la solución encontrada.

Tabla 7.5: *Resumen de los resultados obtenidos en el sistema borroso-genético.*

Algoritmo	**ME**	**MNI**
AG_EST1	0.701	310
AG_EST2	0.719	288
AG_EST3	0.716	339
AG_RAN	0.722	340
AG_DET	0.729	334
AG_IL	0.663	175
AG_SELF	0.748	372
AG_FAMP	0.694	330

Los mejores comportamientos vienen dados por dos AGs Adaptativos, AG_SELF y AG_DET, que han obtenido una evaluación media de 0.748 y 0.729 respectivamente. El peor ha sido, también, un AG Adaptativo; AG_IL, con una evaluación media de 0.663 y una media del número de iteraciones muy baja.

Las tablas 7.6 y 7.7 muestran los resultados obtenidos en cada una de las 30 ejecuciones por cada algoritmo. Para cada experimento, se muestra la evaluación del mejor individuo obtenido y el número de iteración a partir de la cual no ha existido mejora en la solución encontrada.

Tabla 7.6: *Resultados obtenidos en el sistema borroso-genético (I).*

Sistema borroso-genético								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Eval.	Nº It.	Eval.	Nº It.	Eval.	Nº It.	Eval.	Nº It.
1	0.738	401	0.728	194	0.744	478	0.666	108
2	0.718	483	0.748	485	0.725	380	0.666	159
3	0.665	14	0.666	117	0.729	366	0.707	141
4	0.737	195	0.666	32	0.762	485	0.777	463
5	0.665	220	0.665	21	0.753	487	0.723	440
6	0.724	246	0.763	457	0.714	386	0.666	255
7	0.757	478	0.71	470	0.74	363	0.728	427
8	0.706	107	0.761	353	0.666	44	0.733	182
9	0.665	11	0.665	42	0.728	480	0.783	414
10	0.75	450	0.731	271	0.709	294	0.666	51
11	0.692	427	0.666	174	0.728	303	0.729	464
12	0.714	459	0.666	101	0.722	404	0.77	498
13	0.709	483	0.717	253	0.747	433	0.738	487
14	0.665	15	0.74	429	0.699	429	0.666	167
15	0.666	167	0.747	474	0.725	463	0.706	307
16	0.746	489	0.748	384	0.665	60	0.709	338
17	0.764	478	0.767	315	0.684	137	0.755	499
18	0.665	99	0.773	494	0.666	46	0.709	407
19	0.441	491	0.757	368	0.742	445	0.723	497
20	0.722	460	0.721	449	0.665	7	0.716	148
21	0.744	484	0.728	140	0.733	351	0.665	24
22	0.732	397	0.701	487	0.71	499	0.751	472
23	0.755	346	0.733	451	0.69	344	0.694	492
24	0.759	438	0.666	23	0.755	487	0.754	499
25	0.665	28	0.666	163	0.71	457	0.772	387
26	0.665	54	0.666	78	0.75	108	0.724	452
27	0.666	232	0.751	328	0.666	155	0.708	85
28	0.744	405	0.762	219	0.718	491	0.732	427
29	0.666	265	0.743	452	0.732	435	0.74	441
30	0.714	474	0.759	411	0.717	356	0.778	446
MEDIA	**0.701**	**310**	**0.719**	**288**	**0.716**	**339**	**0.722**	**340**

Tabla 7.7: *Resultados obtenidos en el sistema borroso-genético (II).*

Sistema borroso-genético								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Eval.	Nº It.	Eval.	Nº It.	Eval.	Nº It.	Eval.	Nº It.
1	0.773	414	0.66	218	0.727	364	0.713	254
2	0.729	489	0.662	130	0.779	465	0.721	333
3	0.717	479	0.67	151	0.756	495	0.72	446
4	0.717	238	0.677	100	0.787	499	0.665	135
5	0.798	401	0.66	191	0.695	11	0.692	487
6	0.735	344	0.674	144	0.757	365	0.715	458
7	0.718	164	0.59	71	0.799	470	0.658	247
8	0.666	207	0.682	110	0.735	352	0.665	51
9	0.751	336	0.687	350	0.779	486	0.703	427
10	0.666	65	0.696	200	0.72	320	0.666	141
11	0.781	419	0.601	193	0.776	435	0.666	374
12	0.757	462	0.665	80	0.727	492	0.748	417
13	0.752	493	0.658	121	0.718	316	0.696	57
14	0.749	345	0.663	152	0.759	480	0.665	57
15	0.769	408	0.67	186	0.768	298	0.666	314
16	0.754	449	0.695	250	0.754	462	0.706	495
17	0.743	487	0.691	320	0.691	357	0.665	53
18	0.699	115	0.651	190	0.754	172	0.724	494
19	0.713	441	0.702	442	0.735	358	0.729	477
20	0.665	71	0.67	138	0.787	160	0.738	470
21	0.725	192	0.662	142	0.739	486	0.669	489
22	0.666	129	0.623	140	0.711	241	0.689	367
23	0.759	305	0.661	143	0.773	437	0.666	371
24	0.73	483	0.665	79	0.695	250	0.675	499
25	0.742	346	0.66	175	0.739	450	0.666	374
26	0.753	453	0.661	196	0.741	370	0.748	417
27	0.757	330	0.65	176	0.749	432	0.712	319
28	0.701	491	0.694	148	0.772	376	0.665	57
29	0.665	22	0.665	159	0.743	327	0.666	314
30	0.724	455	0.651	163	0.788	452	0.769	500
MEDIA	**0.729**	**334**	**0.663**	**175**	**0.748**	**372**	**0.694**	**330**

7.5.4. Comparación. Análisis estadísticos

Para comparar los resultados obtenidos por AG_SGA con el resto de algoritmos se han realizado diversos tests t-Student; previamente, la prueba de Kolmogorov-Smirnov ha determinado que todos los grupos de datos siguen una distribución normal (p-valor[1]>0.05), como se muestra en la figura 7.10.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	.70063	.71933	.71647	.72180	.72913
	Desviación típica	.060812	.039272	.029463	.036810	.036659
Diferencias más extremas	Absoluta	.246	.213	.133	.135	.124
	Positiva	.149	.213	.123	.135	.124
	Negativa	-.246	-.154	-.133	-.101	-.106
Z de Kolmogorov-Smirnov		1.345	1.165	.730	.741	.680
Sig. asintót. (bilateral)		.054	.132	.661	.643	.744

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	.66387	.74843	.69487	.77780
	Desviación típica	.024956	.029513	.031881	.018449
Diferencias más extremas	Absoluta	.207	.088	.225	.112
	Positiva	.103	.065	.225	.071
	Negativa	-.207	-.088	-.141	-.112
Z de Kolmogorov-Smirnov		1.134	.480	1.231	.616
Sig. asintót. (bilateral)		.153	.975	.097	.843

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Figura 7.10: *Resultados de la prueba Kolmogorov-Smirnov en el sistema borroso-genético.*

La tabla 7.8 muestra el resultado obtenido en cada t-Student, donde se ha indicado con un signo + cuando ha existido una mejora en el comportamiento de AG_SGA con respecto al algoritmo, un signo - si no ha existido mejora y, un signo $\approx$ cuando no han existido diferencias significativas. El valor utilizado para determinar la significación ha sido el estándar, p-valor<0.05.

[1] p-valor se identifica como *Sig. asintót. (bilateral)* en la prueba de Kolmogorov-Smirnov

Tabla 7.8: *Resultados de los tests estadísticos en el sistema borroso-genético.*

Algoritmo	**ME**	**MNI**
AG_EST1	0.700 +	310 -
AG_EST2	0.719 +	288 -
AG_EST3	0.716 +	339 -
AG_RAN	0.721 +	340 -
AG_DET	0.729 +	334 -
AG_IL	0.663 +	175 -
AG_SELF	0.748 +	372 ≈
AG_FAMP	0.692 +	330 -
AG_SGA	0.777	438

Al realizar el análisis estadístico se ha contrastado que la media de las evaluaciones obtenida con AG_SGA es significativamente mayor que la media obtenida con el resto de algoritmos (p-valor[2]<0.001), de los cuales, AG_SELF ha tenido el mejor comportamiento. Puede apreciarse que la mejora de AG_SGA con respecto a AG_SELF ha sido del 2.9 % en la ME, mientras que en la MNI no han existido diferencias. Por tanto, también en este escenario podemos afirmar que el comportamiento de AG_SGA es el más efectivo.

Las siguientes figuras muestran los resultados de los tests t-Student en cada caso.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.70063	.060812	.011103

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	10.285	.002	6.651	58	.000	.077167	.011602	.053942	.100391
	No se han asumido varianzas iguales			6.651	34.293	.000	.077167	.011602	.053595	.100738

Figura 7.11: *Prueba t-Student AG_SGA vs. AG_EST1 en el sistema borroso-genético.*

[2]p-valor se identifica como *Sig. (bilateral)* en la prueba t-Student

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.71933	.039272	.007170

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	24.631	.000	7.380	58	.000	.058467	.007922	.042609	.074324
	No se han asumido varianzas iguales			7.380	41.206	.000	.058467	.007922	.042471	.074463

Figura 7.12: *Prueba t-Student AG_SGA vs. AG_EST2 en el sistema borroso-genético.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.71647	.029463	.005379

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	5.433	.023	9.664	58	.000	.061333	.006347	.048629	.074038
	No se han asumido varianzas iguales			9.664	48.712	.000	.061333	.006347	.048577	.074089

Figura 7.13: *Prueba t-Student AG_SGA vs. AG_EST3 en el sistema borroso-genético.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.72180	.036810	.006721

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	10.587	.002	7.449	58	.000	.056000	.007517	.040952	.071048
	No se han asumido varianzas iguales			7.449	42.705	.000	.056000	.007517	.040837	.071163

Figura 7.14: *Prueba t-Student AG_SGA vs. AG_RAN en el sistema borroso-genético.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.72913	.036659	.006693

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	10.353	.002	6.495	58	.000	.048667	.007493	.033668	.063665
	No se han asumido varianzas iguales			6.495	42.804	.000	.048667	.007493	.033554	.063779

Figura 7.15: *Prueba t-Student AG_SGA vs. AG_DET en el sistema borroso-genético.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.66387	.024956	.004556

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.104	.748	20.107	58	.000	.113933	.005666	.102591	.125276
	No se han asumido varianzas iguales			20.107	53.408	.000	.113933	.005666	.102570	.125296

Figura 7.16: *Prueba t-Student AG_SGA vs. AG_IL en el sistema borroso-genético.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.74843	.029513	.005388

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	6.591	.013	4.621	58	.000	.029367	.006354	.016647	.042087
	No se han asumido varianzas iguales			4.621	48.663	.000	.029367	.006354	.016595	.042139

Figura 7.17: *Prueba t-Student AG_SGA vs. AG_SELF en el sistema borroso-genético.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	.77780	.018449	.003368
	2.00	30	.69487	.031881	.005821

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	14.059	.000	12.332	58	.000	.082933	.006725	.069472	.096395
	No se han asumido varianzas iguales			12.332	46.465	.000	.082933	.006725	.069400	.096466

Figura 7.18: *Prueba t-Student AG_SGA vs. AG_FAMP en el sistema borroso-genético.*

7.6. Conclusiones

En este capítulo se ha presentado el segundo escenario donde ha sido aplicado el sistema de optimización de parámetros propuesto: un AG con codificación híbrida, binaria y real, de un sistema borroso-genético basado en el enfoque de Pittsburgh.

En primer lugar, se ha descrito la estructura del sistema borroso-genético utilizado, así como su integración con el sistema de optimización propuesto.

Posteriormente, se han presentado los resultados obtenidos. Con el sistema de optimización propuesto se han hallado los valores de los parámetros así como la forma en la que varían durante la ejecución del AG del sistema borroso-genético. Seguidamente, se ha comprobado el comportamiento del AG con dichos valores (a este algoritmo se le ha denominado AG_SGA) y se han comparado los resultados, mediante tests estadísticos, con los obtenidos al utilizar otros AGs Adaptativos así como al utilizar los parámetros estáticos recomendados en la literatura.

Con los resultados obtenidos, podemos concluir que AG_SGA también ha sido el algoritmo más efectivo, mejorando en un 2.9 % a AG_SELF, que ha sido el mejor del resto de algoritmos con los que se ha comparado.

Capítulo 8

AG real. Verificación del funcionamiento de protocolos

8.1. Introducción

En este capítulo se describe el último escenario donde se ha aplicado el sistema de optimización propuesto, así como los experimentos realizados y los resultados obtenidos. Se trata de un sistema integrado por un AG con codificación real y un simulador de redes de comunicaciones, con el objetivo de comprobar el funcionamiento de un protocolo de comunicaciones en una determinada red. Tradicionalmente, se utilizan modelos estadísticos como patrones de carga de tráfico para modelar el comportamiento de una red y comprobar así el funcionamiento de un determinado protocolo. El objetivo del AG es hallar, si existe, una configuración de carga de tráfico con la que minimizar el funcionamiento de la red con respecto a los modelos estadísticos, tratando de descubrir posibles debilidades en el protocolo.

La evolución tecnológica ha dado lugar al incremento de la potencia de computación y de la capacidad de transmisión de los ordenadores, lo cual ha provocado el desarrollo de nuevos servicios y aplicaciones en Internet, y en general en las redes de comunicaciones, que demandan unas mayores prestaciones. Así, las redes de comunicaciones están en continua evolución para tratar de dar respuesta a estas nuevas aplicaciones, que cada vez demanda más el mercado; por tanto, los protocolos que rigen su funcionamiento deben ser diseñados para soportar estas nuevas aplicaciones. Esto es particularmente importante ya que, en muchos casos, las nuevas aplicaciones hacen uso de protocolos que no han sido diseñados para su utilización.

De esta forma, la especificación del protocolo, que consiste en proporcionar un algoritmo distribuido en tiempo real que responda a un entorno compuesto por varios usuarios que quieren comunicarse entre sí [Holzmann, 1991], desempeña un papel crucial en el funcionamiento de la red. Algunos aspectos a tener en cuenta en el diseño de un protocolo de comunicación son:

- Robustez: el protocolo debe funcionar correctamente bajo condiciones anormales. Debe estar preparado para todas las acciones y secuencias de acciones posibles (no sólo las probables).

- Consistencia: el protocolo debe ser estable, determinándose la ausencia de propiedades indeseables como: bloqueos, terminaciones inadecuadas, ciclos improductivos, etc.

- Simplicidad: un protocolo debe ser realizado a partir de un número pequeño de módulos bien diseñados.

- Modularidad: en el protocolo debe existir un jerarquia de las funciones a realizar.

El análisis de estos aspectos se realiza generalmente con las siguientes técnicas:

1. Modelado a través de técnicas de descripción formal, que se utilizan para realizar una especificación formal del protocolo a partir de los requerimientos dados, lo cual requiere modelar el comportamiento del protocolo mediante un lenguaje de especificación. Los principales lenguajes de especificación son: SDL, LOTOS y ESTELLE.

2. Simulación, donde un simulador de redes de comunicaciones permite supervisar las acciones realizadas por un protocolo en una red con una topología y una carga de tráfico específicas. El simulador puede modelar un número de fuentes de tráfico que se programan para generar tráfico según modelos estadísticos. Ciertas fuentes utilizarán el protocolo que se desee analizar, mientras que otras generarán la carga de tráfico. Los modelos estadísticos de Poisson [Hui, 1990] son utilizados frecuentemente como modelos de carga de tráfico para modelar el comportamiento de una red de datos. En [Karagiannis et al., 2004, Cao et al., 2003] los autores indican que el uso del modelo de Poisson puede ser válido para el análisis del tráfico de Internet.

En [Baldi et al., 2000, Karthik et al., 2004] se indica que las anteriores técnicas no son suficientes para proporcionar toda la información sobre el funcionamiento de un determinado protocolo. Los autores argumentan que con las técnicas tradicionales de simulación se utilizan unos patrones de tráfico conocidos e incluso, en muchas ocasiones, el análisis de la peor situación en la red de comunicación se realiza diseñando a mano el patrón de la carga de tráfico. A su vez, advierten que con las técnicas de modelado formal se fuerza a trabajar sobre un modelo simplificado del protocolo. Por estas circunstancias, proponen la utilización de una técnica de simulación mixta, basada en la utilización de un AG para la exploración del espacio de soluciones y la búsqueda de inconsistencias en el protocolo con el fin de realizar su verificación.

De esta forma, el sistema consiste en integrar un AG con un simulador de redes de comunicación, como se muestra en la figura 8.1. En un caso, el AG tiene como objetivo generar las peores condiciones de operación y hallar la configuración de carga de tráfico que minimiza el funcionamiento del protocolo que se desee analizar, dadas algunas especificaciones sobre el ancho de banda a utilizar y la topología de la red. En otro caso, y con el fin de conocer el intervalo de variación del *throughput*[1] en la red, el objetivo del AG es hallar la carga de tráfico que maximiza el funcionamiento en esa misma red. Por tanto, el objetivo no es estudiar el protocolo en sí mismo sino comprobar su funcionamiento en una red bajo una carga de tráfico distinta a las utilizadas frecuentemente y generada por el AG.

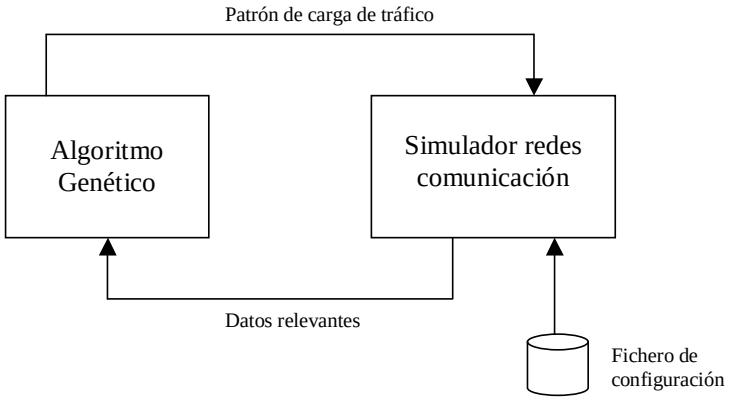

Figura 8.1: *Integración del AG con un simulador de redes de comunicación.*

[1]cantidad de datos por unidad de tiempo entregados sobre un canal de comunicación.

El enfoque mostrado en [Baldi et al., 2000] es muy general y puede ser aplicado a diferentes protocolos utilizando distintos simuladores. En nuestro caso, se ha utilizado el simulador de redes Network Simulator -ns2- [NS2], desarrollado por el Information Sciences Institute de la Universidad de Southern California. Ns2 es un simulador de eventos discreto que simula un amplio rango de tecnologías de redes, tanto cableadas como inalámbricas. Implementa los protocolos TCP y UPD; genera comportamientos de tráfico FTP, Telnet, Web, CBR y VBR; simula mecanismos de gestión de colas en routers Drop Tail, RED y CBQ; soporta diversos algoritmos de enrutamiento; implementa multicasting y algunas de las capas de enlace MAC para simular redes de área local.

El protocolo que se ha seleccionado para comprobar su funcionamiento en una red IP ha sido TCP (Transmission Control Protocol). La topología de la red se describe en el apartado 8.4, donde puede apreciarse la existencia de una serie de conexiones, denominadas *conexiones de prueba*, constituidas por pares de nodos transmisor-receptor de tráfico TCP y sobre las cuales se han realizado las medidas para comprobar el funcionamiento. A su vez, existen otras conexiones, denominadas *conexiones de background*, a través de las cuales la red es cargada con tráfico UDP (User Datagram Protocol). Estas conexiones están constituidas por pares de nodos transmisor-receptor de tráfico UDP. Durante el proceso de simulación, el AG, a partir de la información relevante obtenida de las *conexiones de prueba* (proporcionada por el simulador), genera un patrón de carga de tráfico con el que debe minimizar el funcionamiento del protocolo TCP en la red.

8.2. Estructura del AG real

Para la implementación del AG real de esta parte de la tesis doctoral también se ha utilizado la librería GAlib. Los componentes que se han definido para el AG son los siguientes:

1. Representación genética de las soluciones. Cada individuo de la población representa un patrón o modelo de carga de tráfico en la red, es decir, representa el tráfico generado por las *conexiones de background* durante la simulación. Los individuos se han codificado como un vector de números reales de longitud 4500 genes. Cada gen representa el *delay o retraso* de un paquete UDP de tamaño fijo enviado a la red, es decir, el tiempo que ha de esperar la fuente antes de poder enviar un nuevo paquete a la red después de enviar el actual.

2. Método de formación de la población inicial. La carga de tráfico correspondiente a la población inicial se genera según una distribución de Poisson, donde el tiempo entre paquetes se distribuye de forma exponencial con una media de K=0.67 ms.

3. Sistema de evaluación. Para determinar la evaluación de los individuos el sistema de evaluación mide el *throughput* de las *conexiones de prueba* TCP, es decir, mide las prestaciones que las aplicaciones finales perciben sobre dichas conexiones durante el tiempo de simulación. Todos los bytes recibidos correctamente a nivel TCP pero no entregados a las aplicaciones finales (tales como los paquetes duplicados por el mecanismo de retransmisión del protocolo) no son considerados en las medidas. Por tanto, y dado que el objetivo es minimizar el *throughput*, cuanto mejor sea el individuo hallado por el AG menor debe ser su evaluación. Así, la función de evaluación devuelve el número total de bytes entregados a las aplicaciones finales de las *conexiones de prueba* TCP. La función de evalución utilizada es:

$$f_{Eval}(\vec{x}) = B_{TCP}$$

donde B_{TCP} es el número total de bytes entregados a las aplicaciones finales.

4. Método de selección. Para la selección de los individuos que actúan como padres se han empleado los siguientes métodos: *Proporcional Selection* -selección proporcional-, para el cálculo de las probabilidades de reproducción y, *Tournament Sampling* -torneo- con un tamaño de torneo K=2, como algoritmo de muestreo. El porcentaje de individuos seleccionados viene dado por la P_s.

5. Operadores genéticos. Durante la fase de reproducción se han utilizado los siguientes operadores:

   - Cruce. El cruce de los individuos seleccionados como padres se realiza mediante el operador Uniform Crossover -Cruce uniforme-. Los nuevos individuos tienen una mezcla uniforme de genes de cada uno de los padres.
   - Mutación. Tras el operador de cruce se ha aplicado sobre los nuevos individuos el operador de mutación Random mutation -mutación aleatoria-.

6. Valores de los parámetros de control. El tamaño de la población es de 50 individuos, que es el utilizado en [Baldi et al., 2000] para este mismo problema. El resto de valores de los parámetros se corresponden con los de cada individuo CP_i del SGA. Inicialmente son valores aleatorios (a excepción de CP_1) y evolucionan hacia valores más óptimos al aplicar el SGA propuesto. Los parámetros propuestos en [Baldi et al., 2000] son los siguientes:

Tabla 8.1: *Valores de los parámetros.*

Parámetro	**Valor**
N	50
P_s	0.4
P_c	1.0
P_m	0.01

7. Método de sustitución. La sustitución de los individuos de la población por los individuos generados se realiza mediante el método Replace Worst Strategy (RW) -Reemplazo de los peores-, donde se reemplazan los peores individuos de la población. En los AGs generacionales se ha utilizado una estrategia elitista.

La integración del AG con el simulador ns-2 se muestra en la figura 8.2

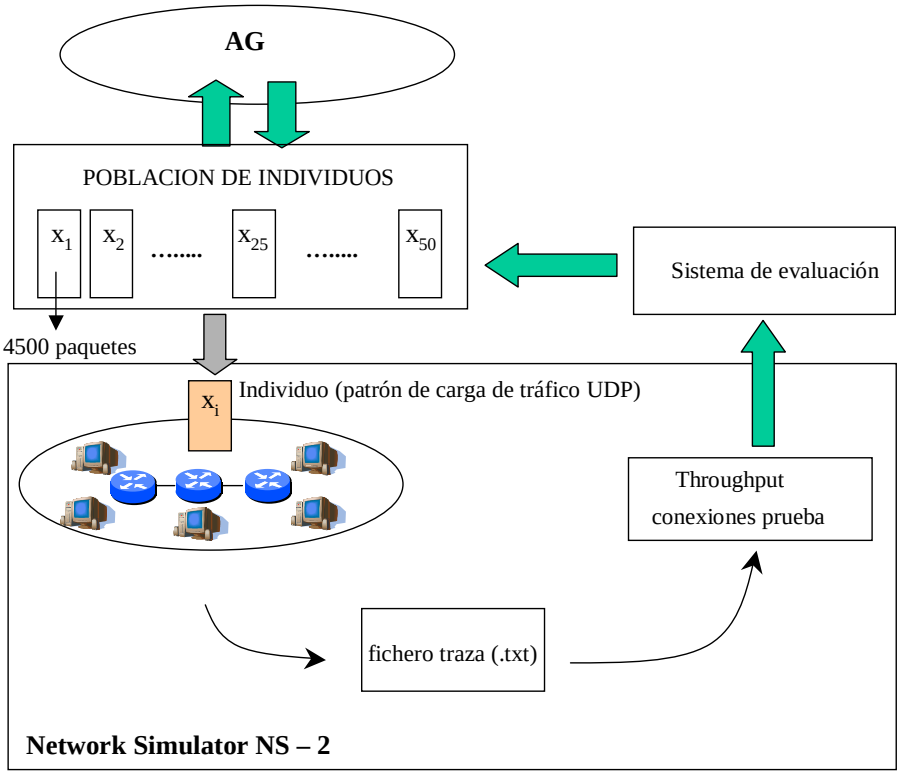

Figura 8.2: *Integración del AG con el simulador NS2.*

8.3. Integración con el sistema de optimización

Al igual que en los dos escenarios anteriores, el objetivo del SGA es hallar los parámetros que optimizan el comportamiento del AG, los cuales van a poder variar su valor en cada una de las iteraciones del algoritmo. La figura 8.3 muestra la integración del SGA con el sistema compuesto por el AG real y el simulador de redes. Como en el primer escenario, los parámetros que se desean optimizar son: P_s, P_c y P_m, por lo que la representación de cada CP_i es:

$$CP_i = \{P_{is}, P_{ic}, P_{im}, a_{i1}, b_{i1}, a_{i2}, b_{i2}, a_{i3}, b_{i3}\}$$

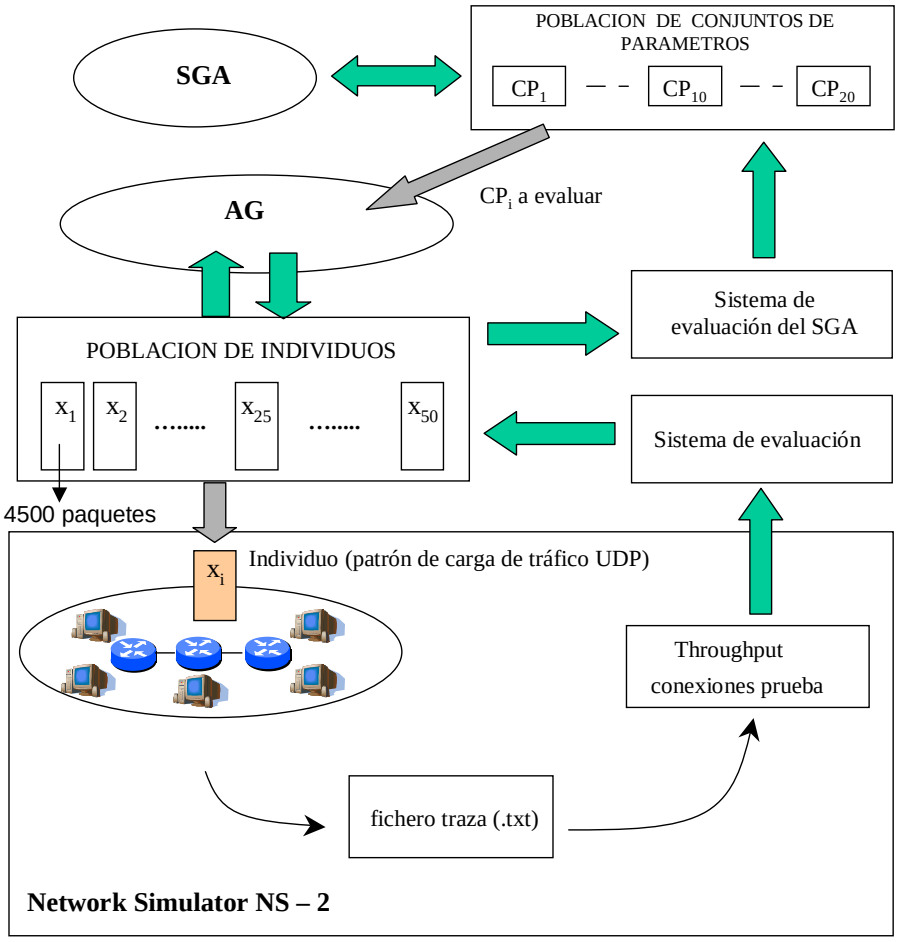

Figura 8.3: *Aprendizaje evolutivo del Conjunto de Parámetros en un AG real.*

En cuanto a la adaptación de los parámetros, también se ha asignado a cada uno una función $p(t)$ diferente. Así, se han utilizado tres funciones $p_{i,\lambda}(t)$, λ=1,...,3 que afectan, de forma independiente, a P_s, P_c y P_m respectivamente. De esta forma, la variación de cada parámetro en cada CP_i durante la ejecución del AG viene dada por:

$$P_{is}(t) = p_0^{is}(1 - \frac{(Ln(t+1))^{(1/a_{i1})}}{(b_{i1}Ln(T+1))^{(1/a_{i1})}}) \quad \begin{matrix} a_{i1} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{ic}(t) = p_0^{ic}(1 - \frac{(Ln(t+1))^{(1/a_{i2})}}{(b_{i2}Ln(T+1))^{(1/a_{i2})}}) \quad \begin{matrix} a_{i2} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

$$P_{im}(t) = p_0^{im}(1 - \frac{(Ln(t+1))^{(1/a_{i3})}}{(b_{i3}Ln(T+1))^{(1/a_{i3})}}) \quad \begin{matrix} a_{i3} \in]0,2] \\ b_{i,1} \in [1,\mathrm{T}] \end{matrix}$$

El proceso de aprendizaje de los parámetros del AG así como de los parámetros $a_{i\lambda}$ y $b_{i\lambda}$ de cada función $p_{i\lambda}(t)$ se realiza de igual forma a la explicada en los dos escenarios anteriores:

En primer lugar, se parte de una población inicial de Conjuntos de Parámetros, CP(0), compuesta por 20 individuos CP_i.

Posteriormente, mientras que el número de iteraciones en el SGA (it_sga) sea inferior a 50:

- se evalua la población CP(it_sga).

  Para evaluar cada CP_i de la población se ejecuta un AG, que utiliza los valores de los parámetros que contiene dicho CP_i. El AG parte de una población inicial P(0) y tras realizarse 100 iteraciones se obtiene una nueva población, P(100). A partir de los individuos obtenidos en P(100) se evalua CP_i.

- Una vez evaluados todos los individuos de CP(it_sga), se incrementa el número de iteraciones en el SGA (it_sga=it_sga+1).

- Seguidamente, se selecciona una nueva población CP(it_sga) a partir de CP (it_sga-1).

- A continuación, se aplican los operadores de cruce y mutación sobre CP(it_sga).

- Por último, se sustituye parte de CP(it_sga-1) por la descendencia generada.

Al final de la ejecución del Sistema Genético Adicional se obtiene una población diferente a la inicial, CP(50), y el CP_i cuya evaluación es la menor, es el que contiene los mejores valores.

En el siguiente apartado se muestra la topología de la red empleada.

8.4. Topología de la red de comunicaciones

La topología de la red utilizada en los experimentos se muestra en la figura 8.4.

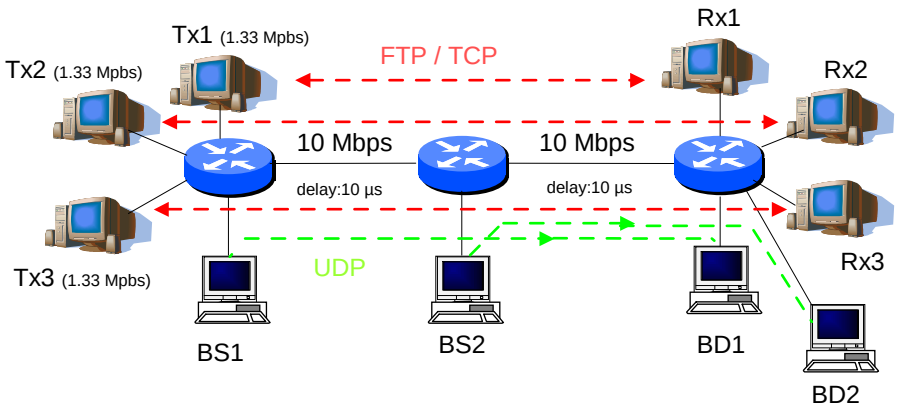

Figura 8.4: *Topología de la red de comunicaciones.*

Como puede apreciarse, existen tres conexiones TCP, desde los nodos transmisores, TX_i, hasta los receptores, RX_i, a través de 3 routers IP. En cada conexión se transfiere un fichero mediante el protocolo de aplicación FTP a una velocidad media de 1.33 Mbps, con un tamaño de paquete de 1024Kbytes. Estas conexiones constituyen las *conexiones de prueba*.

Por otro lado, existen dos fuentes, BS_i (Background Source), que generan tráfico de carga UDP hacia sus respectivos destinos, BD_i (Background Destination), sobre los mismos enlaces que las conexiones TCP. Estas conexiones constituyen las *conexiones de background*.

Cada enlace es full-duplex con una capacidad de 10Mbps y un retardo de 10μs. Los routers se han configurado con un tamaño de cola de 64 paquetes e introducen un retardo de 0.1 ms.

8.5. Resultados

En esta sección se presentan los resultados obtenidos en este escenario. Como ya se ha comentado anteriormente, los modelos de tráfico de Poisson han sido ampliamente utilizados para modelar tráfico en redes de comunicaciones. Así, el objetivo perseguido en este tercer escenario es doble:

a) por un lado, verificar si, verdaderamente, dichos modelos presentan características de tráfico real y, como consecuencia, si los análisis realizados en el simulador pueden ser extrapolados a redes en la realidad.

b) por otro lado, encontrar los parámetros del AG que optimizan su comportamiento, cuyo propósito, en este caso, es hallar el patrón de tráfico con el que minimizar el funcionamiento de un protocolo en la red.

Para alcanzar ambos objetivos se ha procedido de la siguiente forma:

1. En primer lugar, se ha comprobado el funcionamiento de la red mediante la utilización de un modelo de tráfico de poisson como tráfico de *background*.

2. En segundo lugar, se han hallado los parámetros del AG mediante el Sistema Genético Adicional; para ello, se han ejecutado 100 iteraciones en el AG y 50 en el Sistema Genético Adicional.

3. Seguidamente, se ha comprobado el funcionamiento de la red mediante la utilización del modelo de tráfico generado por AG_SGA.

4. A continuación, para poder contrastar el comportamiento de AG_SGA, se ha verificado el funcionamiento de la red mediante la utilización de varios modelos de tráfico generados por otros AGs Adaptativos.

5. Finalmente, se ha comparado el comportamiento en la red al utilizar los distintos patrones de tráfico.

Los siguientes apartados muestran los resultados obtenidos.

8.5.1. Comportamiento de la red con tráfico de *Poisson*

Para comprobar el funcionamiento de la red mediante la utilización de un modelo de poisson se han llevado a cabo 30 simulaciones en el ns-2. En cada una se ha utilizado un patrón de carga distinto que, previamente, ha sido generado aleatoriamente siguiendo una distribución de Poisson. En la figura 8.5 se muestra el throughput obtenido en las conexiones prueba al finalizar cada simulación. Como puede apreciarse, el valor no varía significativamente y está comprendido entre 1.45 Mbps y 1.6 Mbps, con una media de 1.5 Mbps.

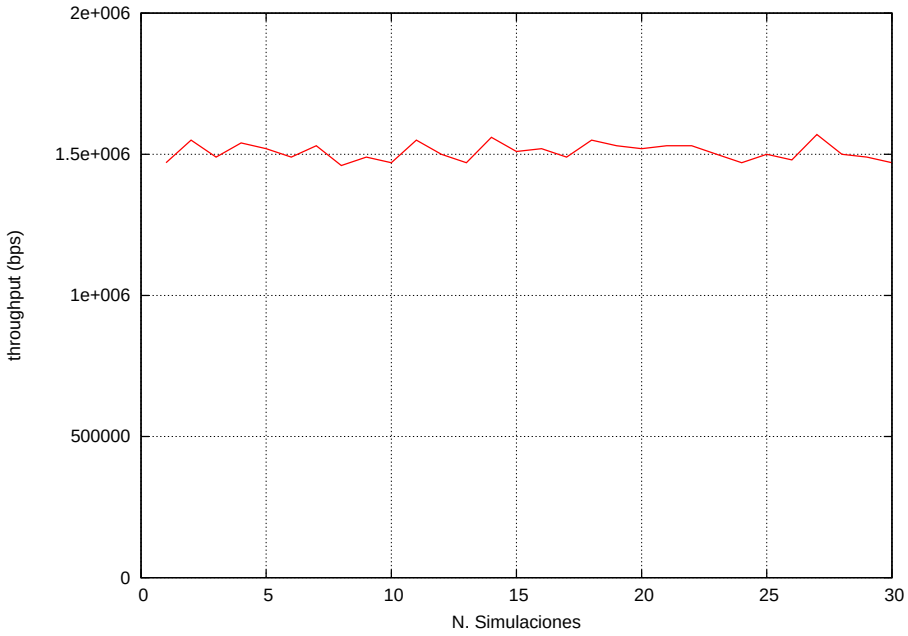

Figura 8.5: *Throughput obtenido mediante un modelo de poisson.*

8.5.2. Parámetros hallados con el sistema

Las probabilidades del AG obtenidas por el Sistema Genético Adicional, que hacen que dicho AG obtenga la peor carga de tráfico en la red y, por tanto, que se minimice el throughput, se muestran en la tabla 8.2. Llama la atención el hecho que la P_m tenga un valor tan bajo, inferior incluso al menor de los valores recomendados en la literatura.

Tabla 8.2: *Valores de los parámetros para la minimización del throughput en la red.*

Parámetro	**Valor**
P_s	0.702
P_c	0.530
P_m	0.00046
a_1 - b_1	0.381 - 38.413
a_2 - b_2	0.753 - 35.661
a_3 - b_3	0.575 - 76.855

También, en este encenario, el sistema ha encontrado como mejor AG a un modelo estacionario, donde el valor inicial de P_s es 0.702. Sin embargo, al contrario que en la mayoría de los casos anteriores, los parámetros no han sido adaptados ya que sus valores iniciales permanecen prácticamente constantes durante la ejecución del algoritmo, como se aprecia en la figura 8.6.

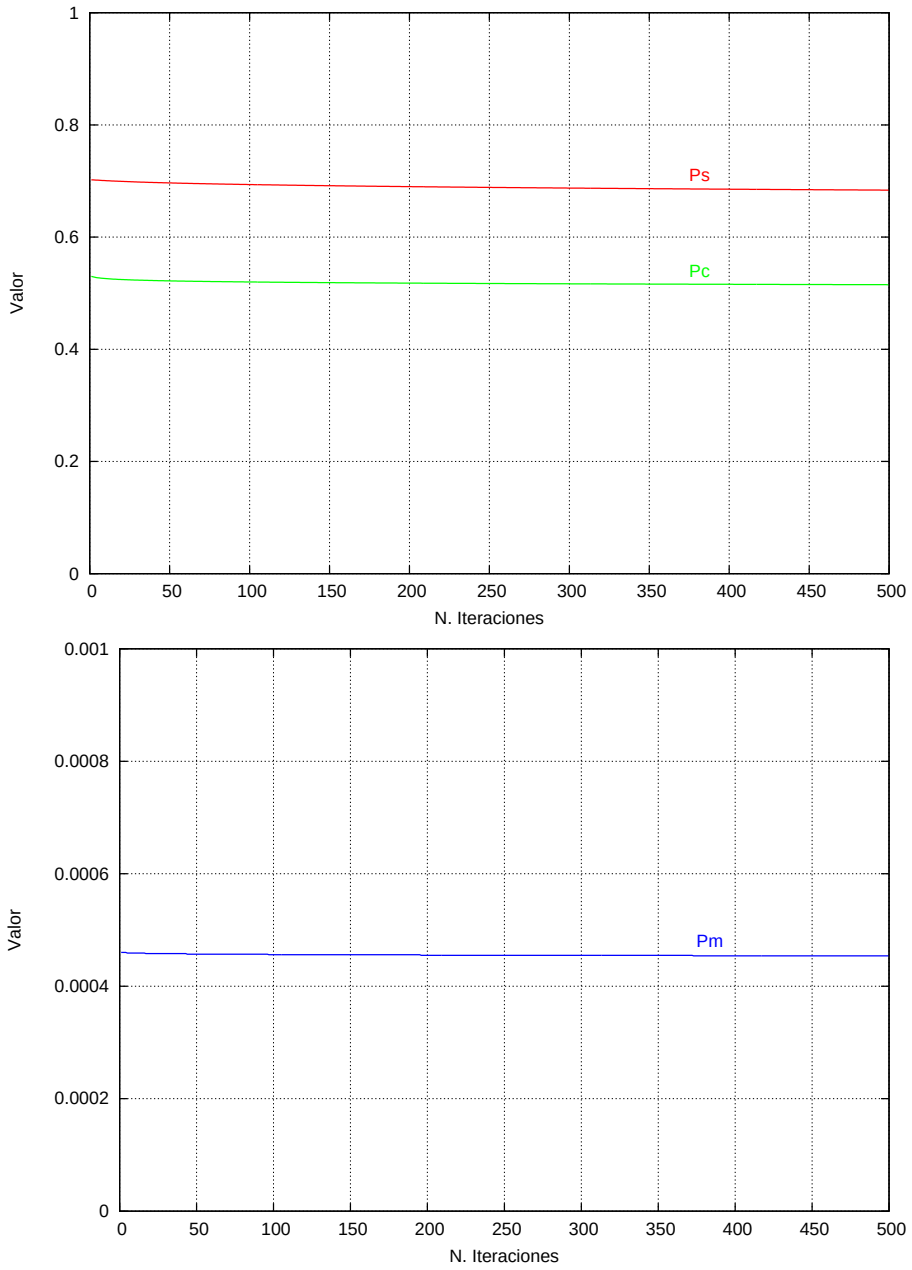

Figura 8.6: *Evolución de P_s, P_c y P_m durante la ejecución del AG para la minimización del throughput.*

8.5.3. Comportamiento del AG con los parámetros hallados

A continuación, se ha comprobado el funcionamiento de la red mediante el modelo de tráfico generado por AG_SGA; para ello, se ha ejecutado el algoritmo 30 veces con un máximo de 500 iteraciones. Los resultados obtenidos se muestran en la tabla 8.3. La primera columna identifica el número de experimento realizado; la segunda, la evaluación del mejor individuo obtenido al final de cada ejecución del AG_SGA (que es el throughput de las conexiones prueba); y la tercera, el número de iteraciones a partir de la cual no ha existido una mejora en la solución encontrada. Cuanto menor es el throughput, peor es la carga de tráfico generada y, por tanto, mejor es el comportamiento del individuo.

Como puede observarse en la figura 8.7, AG_SGA ha conseguido hallar unos patrones de carga de tráfico que hacen que el throughput en las conexiones prueba de la red sea significativamente menor que al emplear el modelo de tráfico de poisson. Sin embargo, para poder evaluar realmente el comportamiento de AG_SGA es necesario conocer el funcionamiento de la red mediante modelos de tráfico generados por otros AGs.

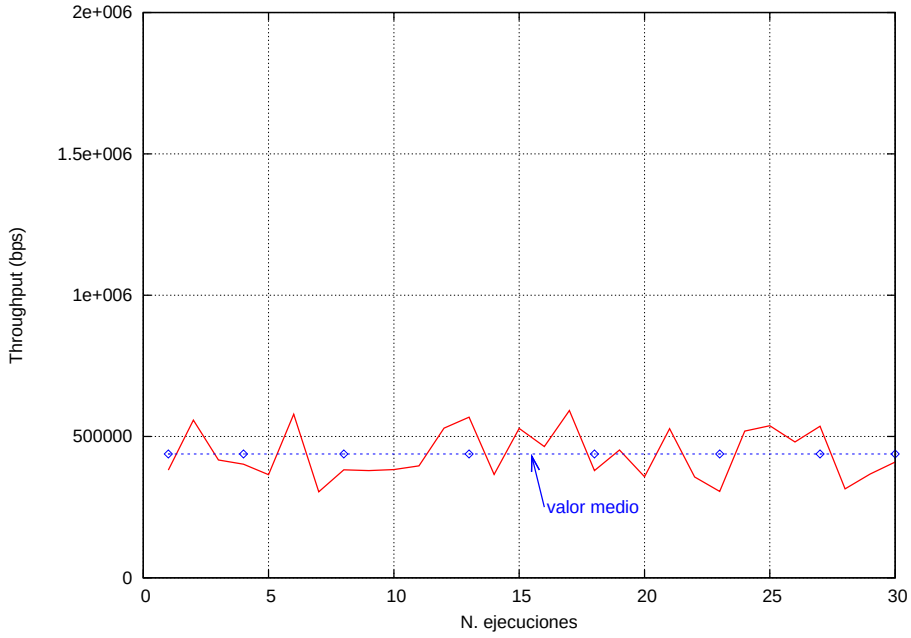

Figura 8.7: *Throughput obtenido mediante el modelo generado por AG_SGA.*

Tabla 8.3: *Throughput obtenido con el modelo generado por AG_SGA.*

Nº Exp.	**Throughput (bps)**	**Nº iteración**
1	381148	287
2	558008	232
3	417087	377
4	401955	120
5	365070	442
6	578815	470
7	304540	492
8	382094	426
9	379256	66
10	383039	498
11	396280	262
12	529635	336
13	568412	342
14	366015	156
15	528689	498
16	464376	482
17	592056	292
18	379256	316
19	452081	435
20	357503	496
21	527744	452
22	356558	221
23	305485	452
24	519231	470
25	538147	297
26	480455	29
27	536255	394
28	314943	305
29	366961	269
30	409521	486
MEDIA	**438020**	**347**

8.5.4. Comportamiento de otros AGs Adaptativos

Al igual que en los dos escenarios anteriores, se han utilizado los principales métodos de adaptación sobre la probabilidad de mutación, los cuales han sido adaptados para que P_m varíe en el intervalo [0.001, 0.01]. Los métodos adaptativos utilizados han sido:

- Control determinístico de P_m (AG_DET), donde P_m ha variado de la siguiente forma:

$$P_m(t) = \ P_m^h - \frac{P_m^h - P_m^l}{T} t \quad 0 \leq \mathrm{t} \leq \mathrm{T}$$

  con P_m^h=0.01, P_m^l=0.001 y T=500.

- Control adaptativo a nivel del individuo de P_m (AG_IL), en el que cada individuo ha tenido asociada una P_m dada por:

$$P_m = \begin{cases} 0.01 & \text{si } f' > \overline{f} \\ 0.01 * \frac{f' - f_{min}}{\overline{f} - f_{min}} & \text{si } f_{min} < f' \leq \overline{f} \\ 0.001 & \text{si } f' = f_{min} \end{cases}$$

  donde f' es la evaluación del individuo, f_{min} es la evaluación del mejor individuo en la población y $\overline{f}$ es la evaluación media de la población.

- Control adaptativo a nivel de la población de P_m (AG_FAMP), donde el controlador borroso se ha disparado cada G=50 iteraciones y P_m ha permanecido fija durante ese tiempo.

- Control auto-adaptativo a nivel del individuo de P_m (AG_SELF), que se ha configurado con los valores de δ=0.001, P_m^l =0.001 y P_m^h=0.01.

- Control aleatorio (AG_RAN).

También se han empleado los siguientes AGs con probabilidades de mutación estáticas:

1. P_m=0.001 (AG_EST1)
2. P_m=0.005 (AG_EST2)
3. P_m=0.01 (AG_EST3)

En todos los casos, los valores de N, P_s y P_c han sido los recomendados en [Baldi et al., 2000]: N=50, P_s=0.4 y P_c=1.0. En la tabla 8.4 se muestra un resumen de los resultados obtenidos por cada algoritmo. La primera columna identifica el algoritmo utilizado; la segunda, muestra la media del menor throughput obtenido al final de cada ejecución del algoritmo (MTh); y la tercera, la media del número de iteraciones a partir de la cual no existe una mejora en la solución encontrada (MNI).

Tabla 8.4: *Throughput medio obtenido en la red por otros AGs.*

Algoritmo.	**MTh (bps)**	**MNI**
AG_EST1	526829	365
AG_EST2	674382	248
AG_EST3	655928	270
AG_RAN	671405	267
AG_DET	668444	324
AG_IL	876471	75
AG_SELF	640813	291
AG_FAMP	656905	264

Como puede apreciarse, todos los AGs, sean adaptativos o no, han hallado unos patrones de carga de tráfico con los que el throughput en las conexiones prueba es significativamente menor que con el modelo de tráfico de poisson.

Por tanto, podemos afirmar que con un modelo de poisson no es posible realizar un análisis de la peor situación en una red de comunicaciones. De esta forma, se comparten las conclusiones extraidas en la tesis doctoral [Aguilar-Igartua, 2000]: que los modelos de poisson no contemplan propiedades del tráfico real en redes de comunicación y asumir este tipo de tráfico puede conllevar a un modelo optimista del comportamiento de una red.

Seguidamente, las tablas 8.5 y 8.6 muestran los resultados obtenidos en cada una de las 30 ejecuciones por cada algoritmo. Para cada experimento, se muestra el througput obtenido por el mejor individuo y el número de iteración a partir de la cual no ha existido mejora en la solución encontrada.

Tabla 8.5: *Resultados obtenidos en la red de comunicaciones (I).*

Red de comunicaciones								
	AG_EST1		AG_EST2		AG_EST3		AG_RAN	
Nº Exp.	Th.(bps)	Nº It.	Th.(bps)	Nº It.	Th.(bps)	Nº It.	Th.(bps)	Nº It.
1	624213	337	522069	422	692309	266	734869	90
2	603405	481	718791	473	742435	181	669610	183
3	523014	398	834175	107	688525	268	716899	218
4	297919	424	741489	362	784995	402	880519	155
5	506936	390	604351	336	429382	367	662044	92
6	572195	407	816206	154	802019	374	835121	246
7	569358	278	545713	68	707441	279	665827	19
8	433165	38	892814	46	520177	328	550442	281
9	484238	494	660152	202	710278	244	701766	309
10	599622	488	698929	347	700821	288	516394	411
11	368853	112	702712	131	552333	78	653532	337
12	535309	358	754730	278	557062	74	574086	368
13	512611	390	700821	305	766079	390	819989	232
14	529635	423	682851	145	732977	168	638399	370
15	389660	448	802019	437	560845	209	683797	237
16	565574	398	803910	380	691363	314	824718	258
17	540039	465	535310	71	643128	498	703658	53
18	638399	167	502207	185	736760	172	573141	495
19	552334	189	504099	20	598677	174	738652	298
20	580707	438	559900	465	549496	329	689471	418
21	527743	458	728248	29	680013	186	530581	90
22	660152	254	687580	280	778374	421	378311	496
23	528689	319	519231	121	573141	482	648803	67
24	587327	398	549496	449	607189	131	828501	268
25	660152	396	562737	179	739598	189	663935	137
26	385877	448	743381	313	660152	389	648803	67
27	527743	379	523960	126	600568	136	635562	283
28	352775	371	742182	392	691363	270	543285	263
29	616647	349	781957	451	595839	332	694274	196
30	530581	454	809427	179	584490	163	737164	269
MEDIA	**526829**	**365**	**674382**	**248**	**655928**	**270**	**671405**	**267**

Tabla 8.6: *Resultados obtenidos en la red de comunicaciones (II).*

Red de comunicaciones								
	AG_DET		AG_IL		AG_SELF		AG_FAMP	
Nº Exp.	Th.(bps)	Nº It.	Th.(bps)	Nº It.	Th.(bps)	Nº It.	Th.(bps)	Nº It.
1	545713	444	1070000	360	655423	229	627050	71
2	664881	492	918350	64	648623	434	687580	204
3	648803	305	1020000	208	721628	158	569357	253
4	749055	348	919295	4	732977	120	835121	499
5	685688	114	1030000	66	550442	244	606243	135
6	716899	390	1240000	3	639345	269	802019	412
7	727302	352	905108	145	583203	409	653532	195
8	627996	425	917404	23	540474	281	762296	193
9	655423	399	1060000	36	554225	481	546659	394
10	571249	21	1100000	28	758513	179	666773	372
11	557063	216	884301	97	616646	218	678122	35
12	775537	155	1140000	30	734868	114	536255	421
13	822826	490	1050000	59	712170	54	675285	446
14	637453	113	1170000	16	593948	146	593002	58
15	637453	133	701766	9	406684	380	627050	233
16	583544	129	778374	51	697535	362	648803	232
17	772700	497	757567	8	673902	279	557063	278
18	679068	383	732031	37	668320	302	668664	226
19	885247	339	807694	21	563988	396	730140	207
20	597731	471	1046980	18	693751	251	555171	489
21	599623	433	798236	117	627562	376	575978	284
22	630833	266	1006310	7	603636	453	645965	403
23	738652	322	675285	60	595184	432	729194	374
24	750947	234	707441	212	674284	331	696092	108
25	683797	315	786886	148	596415	417	616646	99
26	636508	495	544767	79	704291	184	827555	294
27	608134	480	842687	42	650692	414	586382	88
28	606243	417	107632	28	726908	192	719736	292
29	583544	444	736497	187	673906	423	554225	156
30	673393	91	839517	78	624855	215	729194	463
MEDIA	**668444**	**324**	**876471**	**75**	**640813**	**291**	**656905**	**264**

8.5.5. Comparación. Análisis estadisticos

Para determinar si el algoritmo AG_SGA ha generado la peor carga de tráfico se han realizado diversos tests t-Student. Previamente, los resultados de la prueba de Kolmogorov-Smirnov han señalado que todos los grupos de datos de los algoritmos siguen una distribución normal (p-valor[2]>0.05), como se aprecia en la figura 8.8.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_EST1	AG_EST2	AG_EST3	AG_RAN	AG_DET
N		30	30	30	30	30
Parámetros normales[a,b]	Media	526829	674382	655928	671405	668444
	Desviación típica	92307	116632	91024	108681	81585
Diferencias más extremas	Absoluta	.184	.164	.140	.137	.116
	Positiva	.098	.164	.104	.101	.116
	Negativa	-.184	-.129	-.140	-.137	-.066
Z de Kolmogorov-Smirnov		1.005	.899	.766	.753	.637
Sig. asintót. (bilateral)		.265	.394	.600	.622	.812

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Prueba de Kolmogorov-Smirnov para una muestra

		AG_IL	AG_SELF	AG_FAMP	AG_SGA
N		30	30	30	30
Parámetros normales[a,b]	Media	876471	640813	656905	438021
	Desviación típica	221401	75002	83732	88740
Diferencias más extremas	Absoluta	.115	.076	.077	.166
	Positiva	.059	.072	.077	.166
	Negativa	-.115	-.076	-.075	-.153
Z de Kolmogorov-Smirnov		.630	.419	.424	.907
Sig. asintót. (bilateral)		.822	.995	.994	.383

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Figura 8.8: *Resultados de la prueba Kolmogorov-Smirnov en la red de comunicaciones.*

En la tabla 8.7 se muestra el resultado obtenido en cada t-Student, donde se ha indicado con un signo + si ha existido una mejora en el comportamiento de AG_SGA con respecto al algortimo, un signo - si no ha existido mejora y, un signo $\approx$ cuando no han existido diferencias significativas. El valor utilizado para determinar la significación ha sido el estándar, p-valor[3]<0.05.

A la vista de los resultados se pueden realizar las siguientes afirmaciones:

- Se ha contrastado que el throughput medio obtenido en la red al utilizar

[2]p-valor se identifica como *Sig. asintót. (bilateral)* en la prueba de Kolmogorov-Smirnov

[3]p-valor se identifica como *Sig. (bilateral)* en la prueba t-Student

AG_SGA es significativamente menor que el obtenido con el resto de algoritmos (p-valor <0.001).

- Al emplear el patrón de tráfico generado por AG_SGA, el funcionamiento de la red se ha degradado, en media, un 70.79 % con respecto a la utilización de un modelo de poisson.
- Del resto de algoritmos, AG_EST1 ha sido el que ha tenido el mejor comportamiento; ha conseguido degradar el funcionamiento en un 64.87 % con respecto a un modelo de poisson. El peor algoritmo, de nuevo, ha sido AG_IL que ha conseguido generar unos patrones con los que el throughput, de media, ha disminuido a 876471 bps.
- AG_SGA ha conseguido degradar el throughput un 5.92 % más que AG_EST1 en el mismo número de iteraciones.

Por tanto, AG_SGA es el algoritmo que ha generado, de forma significativa, la peor carga de tráfico en la red.

Tabla 8.7: *Resultados tests estadisticos en la red de comunicaciones.*

Algoritmo.	**MT**	**MNI**
AG_EST1	526829 +	365 ≈
AG_EST2	674382 +	248 +
AG_EST3	655928 +	270 +
AG_RAN	671405 +	267 +
AG_DET	668444 +	324 ≈
AG_IL	876471 +	75 +
AG_SELF	640813 +	291 ≈
AG_FAMP	656905 +	264 +
AG_SGA	438020	347

Las siguientes figuras muestran los resultados de los tests t-Student en cada caso.

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	526829	9.2307E+004	1.6853E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.700	.406	-3.799	58	.000	-8.8809E+004	2.3378E+004	-1.3560E+005	-4.2013E+004
	No se han asumido varianzas iguales			-3.799	57.910	.000	-8.8809E+004	2.3378E+004	-1.3561E+005	-4.2012E+004

Figura 8.9: *Prueba t-Student AG_SGA vs. AG_EST1 en la red de comunicaciones.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	674382	1.1663E+005	2.1294E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	2.980	.090	-8.834	58	.000	-2.3636E+005	2.6757E+004	-2.8992E+005	-1.8280E+005
	No se han asumido varianzas iguales			-8.834	54.148	.000	-2.3636E+005	2.6757E+004	-2.9000E+005	-1.8272E+005

Figura 8.10: *Prueba t-Student AG_SGA vs. AG_EST2 en la red de comunicaciones.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	655928	9.1024E+004	1.6619E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.023	.881	-9.389	58	.000	-2.1791E+005	2.3209E+004	-2.6437E+005	-1.7145E+005
	No se han asumido varianzas iguales			-9.389	57.963	.000	-2.1791E+005	2.3209E+004	-2.6437E+005	-1.7145E+005

Figura 8.11: *Prueba t-Student AG_SGA vs. AG_EST3 en la red de comunicaciones.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	671405	1.0868E+005	1.9842E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	.002	.964	-9.111	58	.000	-2.3338E+005	2.5617E+004	-2.8466E+005	-1.8211E+005
	No se han asumido varianzas iguales			-9.111	55.770	.000	-2.3338E+005	2.5617E+004	-2.8471E+005	-1.8206E+005

Figura 8.12: *Prueba t-Student AG_SGA vs. AG_RAN en la red de comunicaciones.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	668444	8.1585E+004	1.4895E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	1.524	.222	-10.470	58	.000	-2.3042E+005	2.2008E+004	-2.7448E+005	-1.8637E+005
	No se han asumido varianzas iguales			-10.470	57.595	.000	-2.3042E+005	2.2008E+004	-2.7448E+005	-1.8636E+005

Figura 8.13: *Prueba t-Student AG_SGA vs. AG_DET en la red de comunicaciones.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	876471	2.2140E+005	4.0422E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	9.499	.003	-10.068	58	.000	-4.3845E+005	4.3548E+004	-5.2562E+005	-3.5128E+005
	No se han asumido varianzas iguales			-10.068	38.083	.000	-4.3845E+005	4.3548E+004	-5.2660E+005	-3.5030E+005

Figura 8.14: *Prueba t-Student AG_SGA vs. AG_IL en la red de comunicaciones.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	640813	7.5002E+004	1.3693E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	3.458	.068	-9.560	58	.000	-2.0279E+005	2.1213E+004	-2.4526E+005	-1.6033E+005
	No se han asumido varianzas iguales			-9.560	56.433	.000	-2.0279E+005	2.1213E+004	-2.4528E+005	-1.6030E+005

Figura 8.15: *Prueba t-Student AG_SGA vs. AG_SELF en la red de comunicaciones.*

Estadísticos de grupo

	grupo	N	Media	Desviación típ.	Error típ. de la media
datos	1.00	30	438021	8.8740E+004	1.6202E+004
	2.00	30	656905	8.3732E+004	1.5287E+004

Prueba de muestras independientes

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
									95% Intervalo de confianza para la diferencia	
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	Inferior	Superior
datos	Se han asumido varianzas iguales	1.000	.321	-9.826	58	.000	-2.1888E+005	2.2275E+004	-2.6347E+005	-1.7430E+005
	No se han asumido varianzas iguales			-9.826	57.805	.000	-2.1888E+005	2.2275E+004	-2.6348E+005	-1.7429E+005

Figura 8.16: *Prueba t-Student AG_SGA vs. AG_FAMP en la red de comunicaciones.*

8.5.6. Maximización del throughput en la red

En los apartados anteriores se ha comprobado como los AGs son capaces de encontrar un patrón de carga de tráfico con el que minimizar el throughput de las conexiones prueba, de forma significativa, con respecto al modelo de poisson. En este punto, surgió la duda de si un AG podría también hallar un patrón de carga de tráfico con el que maximizar el throughput y superar a los modelos de poisson que, como hemos indicado anteriormente, conllevan un modelo optimista del comportamiento de una red.

Para llevar a cabo este experimento se ha configurado un AG con los parámetros recomendados en [Baldi et al., 2000]: N=50, P_s=0.4, P_c=0.4 y P_m=0.01. Se puede decir que se trata del algoritmo AG_EST3, pero configurado para un problema de maximización. Al ejecutar el algoritmo 30 veces, se ha obtenido un throughput medio de 1.64Mbps, como puede observarse en la tabla 8.8 y en la figura 8.17, lo que supone un aumento de 149 Kbps con respecto al modelo de poisson.

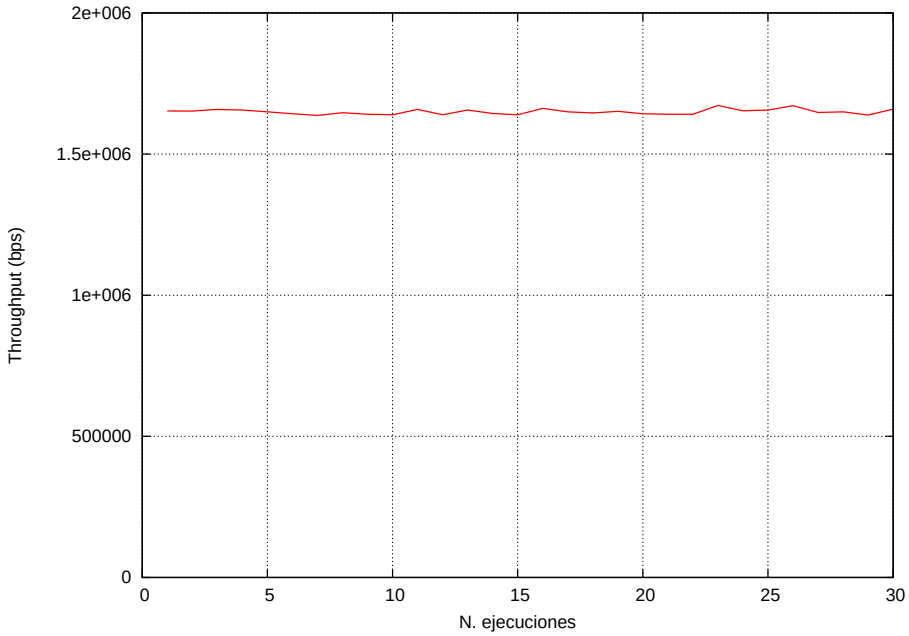

Figura 8.17: *Máximo throughput obtenido mediante el modelo generado por AG_EST3.*

Tabla 8.8: *Máximo throughput obtenido por AG_EST3.*

Nº Exp.	**Max. Throughput (bps)**	**Nº iteración**
1	1652800	364
2	1652270	79
3	1657950	498
4	1656060	480
5	1649440	198
6	1642820	460
7	1637140	281
8	1646600	493
9	1640920	364
10	1639030	369
11	1657950	250
12	1639030	497
13	1656060	100
14	1643760	263
15	1639030	300
16	1661730	405
17	1649440	462
18	1645650	430
19	1651330	375
20	1642820	451
21	1640920	212
22	1640920	408
23	1672140	358
24	1653220	226
25	1656060	446
26	1671190	366
27	1647540	258
28	1649440	160
29	1638090	89
30	1658890	477
MEDIA	**1649675**	**337**

A su vez, también surgió la duda de si AG_SGA (con unas nuevas probabilidades previamente obtenidas por el Sistema Genético Adicional) sería capaz de hallar un throughput mayor al obtenido por el modelo de poisson y por AG_EST3. En este caso, las probabilidades encontradas por el Sistema Genético Adicional son las que se muestran en la tabla 8.9. Como puede observarse, difieren a las obtenidas para el problema de la minimización.

Tabla 8.9: *Valores de los parámetros para la maximización del throughput en la red.*

Parámetro	**Valor**
P_s	0.351
P_c	0.781
P_m	0.145
a_1 - b_1	0.170 - 19.706
a_2 - b_2	0.511 - 18.330
a_3 - b_3	0.100 - 88.427

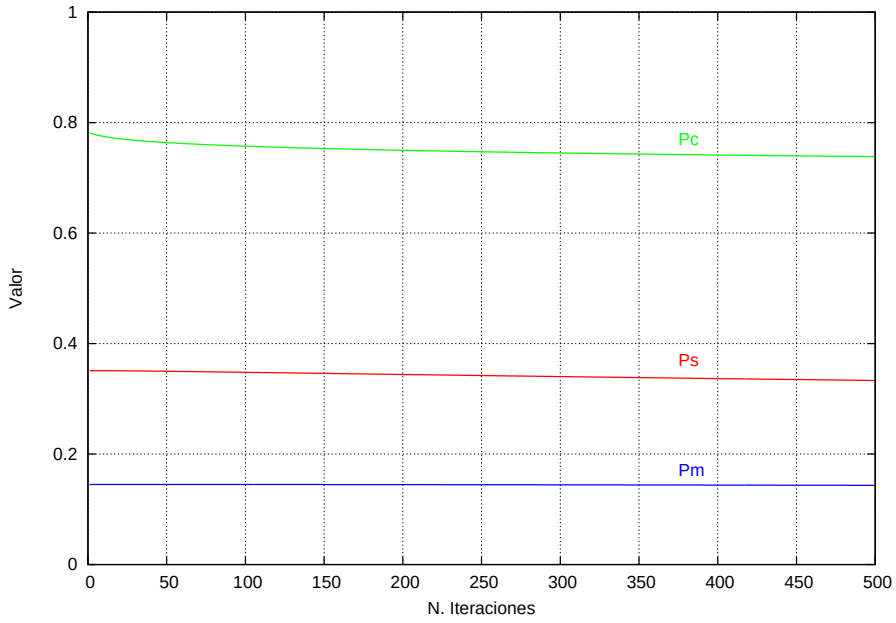

Figura 8.18: *Evolución de P_s, P_c y P_m durante la ejecución de AG_AGS para la maximización del throughput.*

Una vez obtenidas las probabilidades se ha ejecutado AG_SGA 30 veces. La media obtenida ha sido de 2.14 Mbps, como se muestra en la tabla 8.10 y en la figura 8.19, lo que supone un aumento de 645 Kbps con respecto a la media del modelo de poisson y de 496 Kbps con respecto a la media de AG_EST3. Por tanto, AG_SGA ha hallado un patrón de tráfico con el que se ha obtenido el máximo throughput en la red. Así, se ha demostrado que existen otros patrones de tráfico, a priori desconocidos, que hacen que el comportamiento de una red pueda ser aún más optimista que con el modelo de poisson.

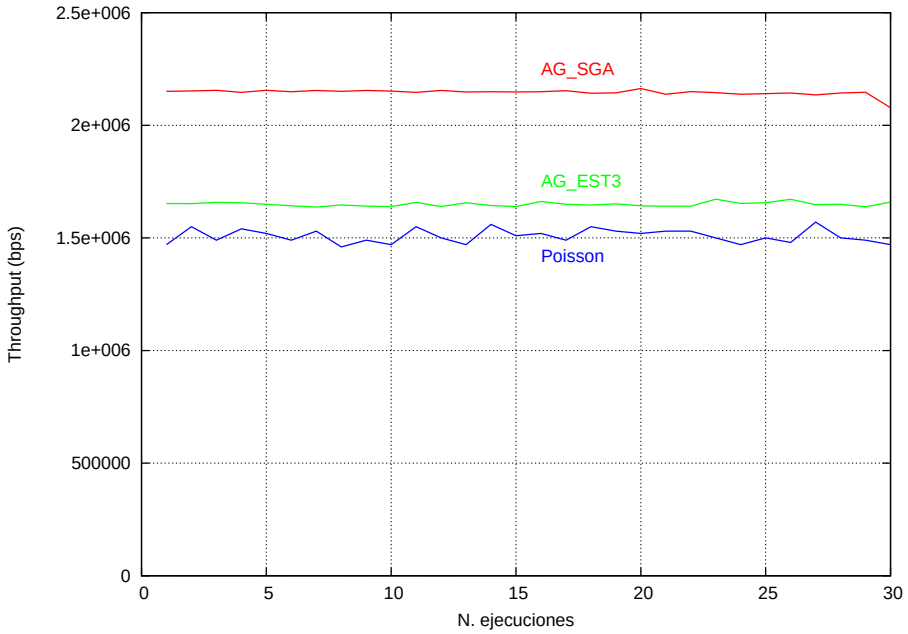

Figura 8.19: *Throughput obtenido mediante el modelo generado por AG_SGA al maximizar.*

Tabla 8.10: *Máximo throughput obtenido por el AG_ SGA.*

Nº Experimento	**Max. Throughput (bps)**	**Nº iteración**
1	2150700	422
2	2152590	257
3	2155430	143
4	2145970	429
5	2155430	411
6	2148810	147
7	2154480	484
8	2150700	242
9	2154480	247
10	2151640	463
11	2145970	499
12	2154480	83
13	2147860	178
14	2148810	401
15	2147860	408
16	2148810	430
17	2153540	444
18	2142190	417
19	2144080	453
20	2162990	470
21	2137460	133
22	2149750	110
23	2145020	88
24	2137460	133
25	2140300	89
26	2143130	77
27	2134620	120
28	2143130	94
29	2146920	144
30	2077870	209
MEDIA	**2145749**	**274**

Para finalizar, en la figura 8.20 se muestra un ejemplo de la evolución del throughput durante las ejecuciones de los algoritmos AG_SGA y AG_EST3. Los datos corresponden a los mejores experimentos de dichos algoritmos, tanto al minimizar como al maximizar. A su vez, también se muestra el resultado al utilizar el modelo de poisson (15 simulaciones y con línea discontinua la media obtenida).

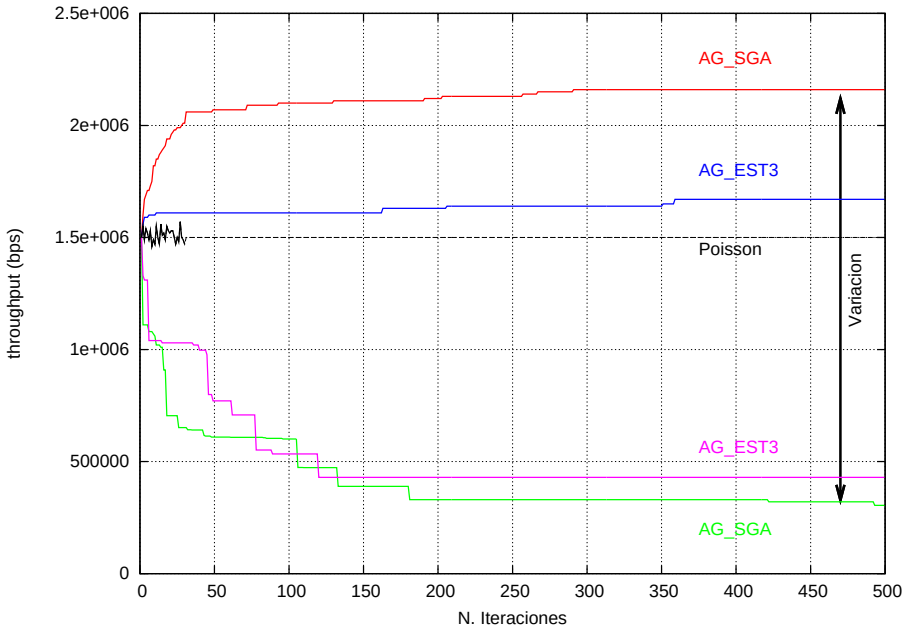

Figura 8.20: *Evolución del throughput en la red de comunicaciones.*

8.6. Conclusiones

En este capítulo se ha presentado el tercer y último escenario donde ha sido aplicado el sistema de optimización de parámetros: un AG con codificación real.

En primer lugar se ha descrito la estructura del AG real, cuyo objetivo ha sido hallar la configuración de carga de tráfico que minimiza el funcionamiento en la red de comunicaciones utilizada. Posteriormente, se ha mostrado cómo se ha integrado el AG real con el simulador de redes de comunicaciones (ns2) y con el sistema de optimización propuesto.

Seguidamente, se han presentado los resultados obtenidos. Se ha comprobado el funcionamiento de la red de comunicaciones al utilizar un modelo de tráfico de poisson como tráfico de *background*. A continuación, con el sistema propuesto, se han hallado los valores de los parámetros así como la forma en la que varían durante la ejecución del AG real. Una vez hallados dichos valores, se ha comprobado el comportamiento del AG (a este algoritmo se le ha denominado AG_SGA) y se han comparado los resultados, mediante tests estadísticos, con los obtenidos al utilizar parámetros estáticos recomendados en la literatura así como al utilizar otros AGs Adaptativos.

A la vista de los resultados obtenidos podemos concluir que AG_SGA ha generado de forma significativa la peor carga de tráfico en la red de comunicaciones. Con el patrón de carga de tráfico hallado por AG_SGA, el funcionamiento de la red se ha degradado, en media, un 70.79 % con respecto a la utilización de un modelo de poisson, y un 5.92 % con respecto a AG_EST1, que ha sido el mejor del resto de algoritmos con los que se ha comparado.

Por último, para finalizar el capítulo, se ha comprobado el comportamiento del AG real al hallar la configuración de carga de tráfico que maximiza el funcionamiento en la red de comunicaciones. También en este caso se ha demostrado que el comportamiento de AG_SGA (con unos nuevos valores previamente obtenidos con el sistema propuesto) ha sido satisfactorio, mejorando en 645Kbps (de media) el funcionamiento en la red al utilizar el tráfico de poisson y, en 496Kbps (de media) al utilizar el AG los valores recomendados en la literatura.

Parte III

Conclusiones y Líneas Futuras

Capítulo 9

Conclusiones y Líneas de Investigación Futuras

Previamente a comentar las conclusiones extraídas del presente trabajo de investigación así como las posibles líneas de investigación futuras, me gustaría, en primer lugar, indicar la principal aportación de esta tesis doctoral y, en segundo lugar, realizar un breve resumen de los resultados obtenidos.

La principal **aportación** del trabajo de investigación llevado a cabo ha sido el desarrollo de un nuevo enfoque para la optimización de los parámetros de control de un AG, basado en la meta-evolución y en la adaptación de dichos parámetros, el cual ha sido aplicado sobre tres tipos de problemas completamente diferentes, obteniendo unos buenos resultados en cada unos de ellos.

Resumen de los resultados obtenidos.

El primer escenario en el que se ha aplicado el sistema propuesto se corresponde con un AG clásico, con codificación binaria, para la resolución de un problema de minimización sobre un conjunto de seis funciones representativas. En este escenario, en cada una de las seis funciones, el AG_SGA (algoritmo que utiliza los parámetros hallados por el sistema propuesto) ha igualado o mejorado el comportamiento del mejor AG con los que se ha comparado.

El segundo escenario en el que se ha aplicado el sistema propuesto ha sido sobre un AG con codificación híbrida, binaria y real, de un sistema borroso-genético basado en el enfoque de Pittsburgh. En este caso, el comportamiento de AG_SGA ha sido el más efectivo de todos los algoritmos comparados, proporcionando una mejora del 2.9 % sobre el algoritmo AG_SELF, que ha sido el mejor del resto de AGs.

Por último, el tercer escenario ha sido sobre un AG con codificación real, el cual se ha integrado con un simulador de redes de comunicaciones. En este caso, el objetivo pretendido por el AG ha sido doble: por un lado, hallar la configuración de carga de tráfico que minimiza el funcionamiento de un protocolo en una red de comunicaciones (minimización del throughput) y, por otro, hallar la configuración de carga de tráfico que maximiza el funcionamiento en esa misma red (maximización de throughput). De esta forma, al conocer el intervalo de variación del throughput, se pueden contrastar las prestaciones dadas en la red en un momento dado. En este tercer escenario, también AG_SGA ha obtenido el mejor comportamiento. En el problema de minimización del throughput en la red de comunicaciones, se ha contrastado que el throughput medio obtenido al utilizar un patrón de tráfico generado por AG_SGA es significativamente menor que el obtenido por el resto de algoritmos. AG_SGA ha conseguido degradar el throughput en la red un 5.92 % más que AG_EST1, que ha sido el mejor de los AGs con los que se ha comparado. Además, al emplear dicho patrón de tráfico generado por AG_SGA, el funcionamiento en la red se ha degradado, en media, un 70.79 % con respecto a la utilización de un modelo de poisson. Por otro lado, en el problema de maximización del throughput, AG_SGA ha conseguido aumentar el throughput en un 42.67 % con respecto a la utilización de un modelo de poisson y un 33.3 % con respecto a AG_EST3, que ha sido configurado con los valores recomendados en la literatura.

Por otra parte, en la mayoría de los casos se ha producido una adaptación de los parámetros del AG_SGA. Únicamente en el escenario con codificación real y en la función f_{RR} del escenario con codificación binario puede considerarse que los parámetros son fijos y sus valores iniciales se mantienen constantes durante la ejecución del algoritmo. En el resto de los casos, los parámetros son adaptados con el método propuesto; en general, P_m ha sido el parámetro que mayor atenuación ha sufrido con respecto a su valor inicial.

En el escenario con codificación binaria, durante la ejecución de AG_SGA, P_m se ha atenuado el 100 % en las funciones f_{Sph} y f_{Ras}, el 28 % en f_{Dec}, el 10 % en f_{One} y el 4 % en f_{Ros} y f_{RR}. A su vez, en el escenario con codificación híbrida, los parámetros que han sufrido una mayor atenuación, con respecto a su valor inicial, también han sido las probabilidades de mutación P_{mv} y P_{mcb}, siendo del 27.5 % y del 25 % respectivamente, mientras que la de P_{mp} ha sido del 6.4 %. En todos estos casos, la atenuación ha sido mucho más acentuada en la etapa inicial de la ejecución del algoritmo, lo cual implica que, en primer lugar, se ha procedido a realizar la exploración de la información, aportando diversidad en la población y, posteriormente, se ha efectuado la explotación de la información, proporcionando convergencia.

9.1. Conclusiones

A partir del trabajo desarrollado en esta tesis se pueden extraer una serie de conclusiones, las cuales se muestran a continuación:

1. El sistema de optimización de parámetros propuesto ha hallado, en cada uno de los escenarios, un conjunto parámetros con los que se ha obtenido el mejor rendimiento en el AG. Al algoritmo que utiliza dichos parámetros se le ha denominado AG_SGA. Con respecto a los AGs con los que se ha comparado, podemos concluir que el comportamiento de AG_SGA ha sido muy robusto, siendo el algoritmo más efectivo y versátil en los distintos escenarios.

2. A la vista de los resultados, se puede afirmar que el comportamiento de un AG basado en un modelo estacionario, donde se reemplaza solamente una parte de la población en cada iteración, es superior al de un AG basado en un modelo generacional, donde se reemplaza la población entera con nuevos individuos en cada iteración. El sistema de optimización propuesto ha encontrado en todos los casos, excepto en la función f_{Dec}, como mejor AG a un modelo estacionario.

3. Se ratifica la existencia de una fuerte dependencia del comportamiento de un AG con los valores de los parámetros que utiliza y con el problema a resolver.

4. Al contrario de lo esperado, no todos los AGs Adaptativos han mejorado el comportamiento de los AGs con parámetros fijos.

   En el escenario con codificación binaria, en todas las funciones, excepto en f_{Ras}, el comportamiento del AG al utilizar parámetros fijos ha igualado al obtenido

con el mejor de los AGs Adaptativos. En el escenario con codificación híbrida, ha mejorado a AG_IL y AG_FAMP, mientras que en el escenario con codificación real el AG con parámetros fijos ha superado a todos los AGs Adaptativos. Llama la atención el mal funcionamiento del algoritmo adaptativo AG_IL en todos los escenarios. Las premisas teóricas de su diseño parecen ser buenas, sin embargo, no se han visto reflejadas en la práctica.

Por tanto, se puede concluir que la adaptación de los parámetros no es el aspecto más influyente en la mejora del comportamiento de un AG, sino la selección de un conjunto de parámetros apropiado, aunque estos permanezcan fijos durante su ejecución.

5. Partiendo de la hipótesis que la probabilidad de mutación es el parámetro más influyente a la hora de determinar el grado de diversidad en la población [Herrera y Lozano, 2003, Rojas et al., 2002], el sistema propuesto ha hallado unas probabilidades de mutación que, en la mayoría de los casos, se sitúan fuera del intervalo recomendado en la literatura, [0.01, 0.001]. Por tanto, si este intervalo se aumentase, por ejemplo, a [0.05, 0.0005], posiblemente el comportamiento de los AGs Adaptativos mejoraría.

6. Por último, es necesario indicar que la aplicación del sistema propuesto a un determinado escenario conlleva un coste computacional adicional. Corresponde a cada usuario valorar qué costes computacionales son aceptables para su problema y si las mejoras introducidas en la solución pueden merecer la pena.

9.2. Líneas de Investigación Futuras

En este apartado se muestran las posibles líneas de investigación que, a juicio del autor, pueden surgir en un futuro a partir del trabajo realizado.

1. Para solucionar el coste computacional adicional que conlleva el sistema propuesto, puede resultar interesante utilizar AGs paralelos o distribuidos. Estos algoritmos, que han sido desarrollados en los últimos años, permiten reducir el tiempo de ejecución, al ejecutarse sobre máquinas distribuidas. A su vez, permiten obtener mejoras en el comportamiento debido a la separación espacial de las soluciones, que evita un convergencia prematura en muchos problemas

[Herrera, 2004]. Como punto de partida para esta línea se propone la referencia [Alba y Tomassini, 2002].

2. Extender la representación de los Conjuntos de Probabilidades, CP_i, para tratar de optimizar, a su vez, los operados genéticos de cruce y mutación. En este sentido, también se podría realizar una adaptación, donde diferentes operadores de cruce o mutación podrían ser utilizados durante la ejecución del AG.

3. Comprobar el comportamiento de los AGs Adaptativos en los escenarios descritos al aumentar el intervalo del dominio de la probabilidad de mutación a [0.05, 0.0005].

4. Finalmente, en relación con el tercer escenario donde ha sido aplicado el sistema propuesto, AG Real - Network Simulator, considero que pueden abordarse las siguientes nuevas vías de investigación:

   - Análisis de la evolución del tamaño de ventana del protocolo TCP cuando se utiliza en la red tráfico generado por los distintos AGs.
   - Análisis del propio tráfico generado por los AGs, comprobando la distribución que siguen los paquetes UDP.
   - Comprobar el comportamiento de otros protocolos de comunicación en la red. Por ejemplo, se podría sustituir la aplicación de transferencia de archivos por una aplicación de voz sobre IP (VoIP), donde en las conexiones de prueba se utilizarían los protocolos RTP (Real Time Protocol), a nivel de aplicación y UDP, a nivel de transporte.
   - Comprobar el comportamiento de otro tipo de redes, como por ejemplo, las redes inalámbricas sin infraestructura, MANET (Mobile and Ad hoc NETworks), o las malladas, redes MESH.

9.3. Publicaciones generadas

1. J.A. Fernández Prieto, J.R. Velasco Pérez. *Design of an Adaptive Genetic Algorithm for Maximizing and Minimizing Throughput in a Computer Network.* Proceedings of the 23rd International Symposium on Computer and Information Sciences (ISCIS08). Estambul, Turquía, octubre 2008.

2. J.A. Fernández Prieto, J.R. Velasco Pérez. *Adaptive Genetic Algorithm control parameter optimization to verify the network protocol performance.* Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU08). Torremolinos, España, junio 2008.

3. J.A. Fernández Prieto, J.R. Velasco Pérez. *Optimización evolutiva de los parámetros de control de un algoritmo genético adaptativo en un sistema borroso-genético.* Actas de la XII Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA07). Salamanca, España, octubre 2007.

4. J.A. Fernández Prieto, J.R. Velasco Pérez. *Parámetros de control dinámicos en un algoritmo genético.* Actas del XII Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF04). Jaén, España, septiembre 2004.

5. J.A. Fernández Prieto, J.R. Velasco Pérez. *Influencia de la población inicial en un algoritmo genético de un sistema borroso-genético.* Actas del XI Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTLYF 02). León, España, septiembre 2002.

6. J.A. Fernández Prieto, J.R. Velasco Pérez. *Method for the optimization of learning systems parameters.* Proceeding of WSES International Conference on Evolutionary Computation (EC01). Puerto de la Cruz, España, febrero 2001.

7. J.A. Fernández Prieto, J.R. Velasco Pérez. *Method for the optimization of learning systems parameters.* Advances in Fuzzy Systems and Evolutionary Computation. Vol 1, pp. 271-276, enero 2001.

8. J.A. Fernández Prieto, J.R. Velasco Pérez. *Application of Genetic Algorithms in the research of the optimum probabilities of the genetic operators.* Proceedings of the 8th International Conference on Information Processing and Mana-

gement of Uncertainty in Knowledge Based Systems (IPMU00). Madrid, España, septiembre 2000.

Bibliografía

M. Aguilar-Igartua. *Contribución al modelado y caracterización de nodos en redes de banda ancha. Aplicación al multiplexor inverso ATM*. PhD thesis, Universidad Politécnica de Cataluña, 2000.

J. T. Alander. On optimal population size of genetic algorithms. In *Proceedings Computer Systems and Software Engineering, 6th Annual European Computer Conference*, pp. 65–70, 1992.

E. Alba y M. Tomassini. Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6:443–462, 2002.

J. Arabas, Z. Michalewicz, y J. Mulawka. Gavaps - a genetic algorithm with varying population size. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, pp. 73–78, 1994.

J. E. Baker. Adaptive selection methods for genetic algorithms. In *Proceeding of the First Int. Conf. on Genetic Algorithms*, pp. 101–111, 1987a.

J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In J. Grefenstette, editor, *Proceedings of the Second Int Conference on Genetic Algorithms and their Applications*, pp. 14–21, 1987b.

M. Baldi, F. Corno, M. Rebaudengo, M. Sonza, y others G. Squillero. *In Telecommunications Optimization: Heuristic and Adaptive Techniques*, chapter GA-based Verification of Network Protocols Performance, pp. 185–198. John Wiley and Sons, Ltd., 2000.

T. Bäck. Self-adaptation in genetic algorithms. In *Proceeding of the first European Conference on Artificial Life*, pp. 11–13, 1991.

T. Bäck. The interaction of mutation rate, selection, and self-adaptation within a genetic algorithm. In Männer y Manderick, editors, *Proceeding of the second Conference Parallel Problem Solving from Nature*, pp. 85–94, 1992.

T. Bäck. *Evolutionary algorithms in Theory and Practice*. Oxford University Press, 1996.

T. Bäck y M. Schütz. Intelligent mutation rate control in canonical genetic algorithms. In R. ZW y M. M, editors, *Foundation of Intelligent Systems 9th Int Symposium*, pp. 158–167, 1996.

J. C. Bean y A. B. Hadj-Alouane. A dual genetic algorithm for bounded integer programs. Technical report, Department of Industrial and Operations Engineering, The University International of Michigan, 1992.

L. Booker. *Genetic Algorithms and Simulated Annealing*, chapter Improving search in genetic algorithms, pp. 61–73. Morgan Kaufmann Publishers, 1987.

M. F. Bramlette. Initialization, mutation and selection methods in genetic algorithms for function optimization. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 100–107, 1991.

J. Cañada-Bago. *Aprendizaje en sistema borrosos mediante evaluación basada en conocimiento. Aplicación en sistemas fotovoltaicos autónomos.* PhD thesis, Universidad Politécnica de Madrid, 2004.

E. Cantú-Paz. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*, chapter 13 Parameter Setting in Parallel Genetic Algorithms, pp. 259–276. 2007.

J. Cao, W. Cleveland, D. Lin, y D. Sun. *Lectures Notes in Statistics*, chapter Internet traffic Tends Toward Poisson and Independent as the Load Increases, pp. 83–109. Springer-Verlag, 2003.

Y. J. Cao y Q. H. Wu. Optimization of control parameters in genetic algorithms: a stochastic approach. *International Journal of Systems Science*, 30(5):551–559, 1999.

V. A. Cicirello y S. F. Smith. Modeling ga performance for control parameter optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 235–242. Morgan Kaufmann Publishers, 2000.

C. A. Coello, D. A. Van-Veldhuizen, y G. B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*, volume 5 of *Genetic Algorithms and Evolutionary Computation*. Kluwer Academic Publishers, 2002.

O. Cordón, M. J. del Jesús, F. Herrera, y M. Lozano. Mogul: A methodology to obtain genetic fuzzy rule-based systems under the iterative rule learning approach. *International Journal of Intelligent Systems*, 14:1123–1153, 1999.

O. Cordón, F. Gomide, F. Herrera, F. Hoffmann, y L. Magdalena. Ten years of genetic fuzzy systems: current framework and new trends. *Fuzzy Sets and Systems*, 141:5–31, 2004.

O. Cordón, F. Herrera, F. Hoffmann, y L. Magdalena. *Genetic Fuzzy Systems: Evolutionary tuning and learning of fuzzy knowledge bases*, volume 19 of *Advances in fuzzy systems - Applications and teory*. World Scientific Publishing, 2001.

C. Darwin. *On the Origin of Species by Means of Natural Selection*. John Murray, London, 1859.

L. Davis. Adapting operator probabilities in genetic algorithms. In *Proceeding of the Third Int. Conf. on Genetic Algorithms*, pp. 61–69, San Mateo, 1989. Morgan Kaufmann Publishers.

L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

K. Deb. *Multi-Objective using Evolutionary Algorithms*. John Wiley and Sons, 2001.

K. Deb. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*, chapter 12 Evolutionary Multi-Objective Optimization Without Additional Parameters, pp. 241–257. 2007.

K. DeJong. *An analysis of the behaviour of a class of genetic adaptive systems*. PhD thesis, University of Michigan, 1975.

K. DeJong. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*, chapter 1 Parameter setting in EAs: a 30 year perspective, pp. 1–18. 2007.

A. E. Eiben y J. K. V. der Hauw. Adaptive penalties for evolutionary graph-coloring. In *Proceedings of Evolution Artificielle*, number 1363 in LNCS, pp. 95–106, 1997.

A. E. Eiben, R. Hinterding, y Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3:124–141, 1999.

A. E. Eiben, Z. Michalewicz, M. Schoenauer, y J. E. Smith. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*, chapter 2 Parameter Control in Evolutionary Algorithms, pp. 19–46. 2007.

A. E. Eiben y Z. Ruttkay. Self-adaptivity for constraint ter. satisfaction: Learning penalty functions. In *Proceedings of the 3rd IEEE Conference on Evolutionary Computation*, pp. 258–261. IEEE press, 1996.

L. J. Eshelman. The chc adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination. In *Proceedings Foundation of Genetic Algorithms*, pp. 265–283. Morgan Kaufmann Publishers, 1991.

L. J. Eshelman, A. Caruana, y J. D. Schaffer. Biases in the crossover landscape. In *Proceeding of the Third Int. Conf. on Genetic Algorithms*, pp. 86–91, San Mateo, 1989. Morgan Kaufmann Publishers.

L. J. Eshelman y J. D. Schaffer. Real-coded genetic algorithms and interval-schemata. In *Proceedings Foundation of Genetic Algorithms*, pp. 187–202. Morgan Kaufmann Publishers, 1993.

T. C. Fogarty. Varying the probability of mutation in the genetic algorithm. In *Proceeding of the Third International Conference on Genetic Algorithms and Their Applications*, pp. 104–109. Morgan Kaufmann Publishers, 1989.

D. B. Fogel. *System Identification through Simulated Evolution: A Machine Learning Approach to Modeling.* Gin Press, 1991.

L. J. Fogel. Autonomous automata. *Industrial Research*, 4:14–19, 1962.

L. J. Fogel, A. J. Owens, y M. J. andWalsh. *Artificial Intelligence through Simulated Evolution.* John Wiley Sons, 1966.

S. Forrest y M. Mitchell. Relative building block fitness and the building block hypothesis. In *Proceedings Foundations of Genetic Algorithms*, pp. 109–126. Morgan Kaufmann Publishers, 1993.

GAlib. A c++ library of genetic algorithm - disponible en: http://lancet.mit.edu/ga.

D. E. Goldberg. *Genetic Algorithms in search, optimization and Machine Learning.* Addison-Wesley, New York, 1989.

D. E. Goldberg, K. Deb, y B. Korb. Do not worry, be messy. In *Proceedings of the 4th International Conference on Genetic Algorithms*, pp. 24–30. Morgan Kaufmann Publishers, 1991.

D. E. Goldberg, B. Korb, y K. Deb. Messy genetic algorithms: motivation, analysis, and first results. *Complex Systems*, 3:493–530, 1989.

J. J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 16(1):122–128, 1986.

F. Herrera. Sistemas difusos evolutivos. In *Actas del XII congreso español sobre tecnologías y lógica fuzzy*, pp. 1–8, 2004.

F. Herrera. Genetic fuzzy systems: Status, critical considerations and future directions. *International Journal of Computational Intelligence Research*, 1(1):59–67, 2005.

F. Herrera y M. Lozano. *Genetic Algorithms and Soft Computing*, chapter Adaptaton of Genetic Algorithm Parameters Based on Fuzzy Logic Controllers, pp. 95–125. Physica-Verlag, 1996.

F. Herrera y M. Lozano. Fuzzy adaptive genetic algorithms: design, taxonomy, and future directions. *Soft Computing*, 7:545–562, 2003.

F. Herrera, M. Lozano, y J. L. Verdegay. Algoritmos geneticos: Fundamentos, extensiones y aplicaciones. *ARBOR CLII*, 597:9–40, 1995a.

F. Herrera, M. Lozano, y J. L. Verdegay. The use of fuzzy connectives to design real-coded genetic algorithms. *Mathware and SoftComputing*, 1(3):239–251, 1995b.

F. Herrera, M. Lozano, y J. L. Verdegay. Dynamic and heuristic fuzzy connectives based crossover operators for controlling the diversity and convergence of real-coded genetic algorithms. *International Journal of Intelligent Systems*, 11:1013–1041, 1996.

F. Herrera y L. Magdalena. Genetic fuzzy systems. *Fuzzy Structures. Current Trends. Lecture Notes of the Tutorial: Genetic Fuzzy Systems*, 13:93–121, 1997.

J. Hesser y R. Männer. Towards an optimal mutation probability in genetic algoritms. In *Parallel problem solving from nature*, volume 496 of *Lecture Notes in Computer Science*, pp. 23–32, 1991.

R. Hinterding, Z. Michalewicz, y T. C. Peachey. Self-adaptive genetic algorithm for numeric functions. In *Proceeding of the 4th Conference on Parallel Problem Solving for Nature*, Lecture Notes in Computer Science, pp. 420–429, Berlin, 1996. Springer.

J. H. Holland. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press (The MIT Press, London, 1992), 1975.

G. J. Holzmann. *Design and validation of computer protocols.* Prentice Hall, 1991.

J. Hui. *Switching and Traffic Theory for Integrated Broadband Networks.* Kluwer Academic Publishers Norwell, 1990.

T. Jansen, K. DeJong, y I. Wegener. On the choice of offspring population size in evolutionary algorithms. *Evolutionary Computation*, 13(4):413–440, 2005.

J. A. Joines y C. R. Houck. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with gas. In *Proceedings of the first IEEE Conference on Evolutionary Computation*, pp. 579–584. IEEE Press, 1994.

B. A. Julstrom. What have you done for me lately adapting operator probabilities in a steady-state genetic algorithm. In *Proceedings of the Sixth Int. Conf. on Genetic Algorithms*, pp. 81–87, San Francisco, 1995. Morgan Kaufmann Publishers.

T. Karagiannis, M. Molle, M. Faloutsos, y A. Broido. A nonstationary poisson view of internet traffic. In *Proceedings of the Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 3, pp. 1558–1569, Hong Kong, 2004.

S. Karthik, V. Jawajar, B. Chidambararajan, y S. K. Srivatsa. Performance of tcp over satellite networks under severe cross-traffic using ga. *International Journal Mobile Communications*, 2(4):382–394, 2004.

A. Konak, D. W. Coit, y A. E. Smith. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, 91:992–1007, 2006.

J. R. Koza. *Genetic Programming.* Cambridge; MA: MIT Press, 1992.

T.-H. Li, C. B. lucasius, y G. Kateman. Optimization of calibration data with the dynamic genetic algorithm. In *Proceedings Analytica Chimica*, pp. 123–134, 1992.

J. Lis y M. Lis. Self-adapting parallel genetic algorithm with the dynamic mutation probability, crossover rate and population size. In *Proceeding of the first Polish National Conference on Evolutionary Computation*, pp. 324–329, 1996.

F. G. Lobo y C. F. Lima. A review of adaptive population sizing schemes in genetic algorithms. In *Proceedings of the 2005 workshops on Genetic and evolutionary computation*, pp. 228–234, Washington D.C., 2005.

F. G. Lobo y C. F. Lima. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*, chapter 9 Adaptive Population Sizing Schemes in Genetic Algorithms, pp. 185–204. 2007.

L. Magdalena y F. Monasterio. Evolutionary-based learning applied to fuzzy controllers. In *Proceedings of 4th IEEE International Conference on Fuzzy Systems*, volume 3, pp. 1111–1118, Yokohama, Japan, 1995.

L. Magdalena y J. R. Velasco. *Genetic Algorithms and Soft Computing*, chapter Fuzzy Rule-Based Controllers that Learn by Evolving their Knowledge Base, pp. 172–201. Physica-Verlag, 1996.

E. H. Mamdani y S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7:1–13, 1975.

S. Meyer-Nieberg y H.-G. Beyer. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*, chapter 3 Self-Adaptation in Evolutionary Algorithms, pp. 47–75. 2007.

H. Mühlenbein. How genetic algorithms really work: I. mutation and hillclimbing. In *Proceedings of the second Conference on Parallel Problem Solving from Nature*, pp. 15–25, 1992.

H. Mühlenbein y D. Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm i. continuous parameter optimization. *Evolutionary Computation*, 1: 25–49, 1993.

Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, 1992.

Z. Michalewicz y N. Attia. Evolutionary optimization of constrained problems. In *Proceeding of the 3rd Annual Conference on Evolutionary Programming*, pp. 98–108, 1994.

Z. Michalewicz y M. Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4:1–32, 1996.

Y.Ñojima y H. Ishibuchi. Computational efficiency of parallel distributed genetic fuzzy rule selection for large data sets. In *Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU08)*, pp. 1137–1142, June 2008.

NS2. The network simulator - disponible en: http://www.isi.edu/nsnam/ns.

W. Pedryck. *Fuzzy Control and Fuzzy Systems.* Wiley, 1989.

E. Perez, F. Herrera, y C. Hernandez. Finding multiple solutions in job shop scheduling by niching genetic algorithm. *Journal of Intelligent Manufacturing*, 14:223–239, 2003.

N. J. Radcliffe. Forma analysis and random respecful recombination. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 222–229, San Mateo, 1991. Morgan Kaufmmann.

I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Frommann-Holzboog, Stuttgart, 1973.

I. Rojas, J. González, H. Pomares, J. J. Merelo, P. A. Castillo, y G. Romero. Statistical analysis of the main paremeters involved in the design of a genetic algorithm. *IEEE Transtions on Systems, Man and Cybernetic- Part C: Applications and reviews*, 32(1):31–37, 2002.

F. Sahin y G. Abbate. Genetic algorithm parameter optimization: applied to sensor coverage. In *Proceedings of the Digital Wireless Communications VI*, volume 5440, pp. 157–168, 2004.

J. D. Schaffer, R. A. Caruana, L. J. Eshelman, y R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization.

In *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications*, pp. 51–60, San Mateo, June 1989.

N. Schraudolph y R. Belew. Dynamic parameter encoding for genetic algorithms. *Machine Learning*, 9:9–21, 1992.

H. P. Schwefel. *Numerical optimization of computer models*. John Wiley-Sons, New York, 1981.

C. G. Shaefer. The argot strategy: Adaptive representation genetic optimizer technique. In *Proceedings of the 2nd International Conference on Genetic Algorithms*, pp. 50–55. Lawrence Erlbaum Associates, 1987.

A. E. Smith y D. M. Tate. Genetic optimization using a partment penalty function. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pp. 499–503. Morgan Kaufmann, 1993.

R. E. Smith. Adaptively resizing populations: An algorithm and analysis. In *Proceedings of the 5th International Conference on Genetic Algorithms*, page 653. Morgan Kaufmann, 1993.

M. Srinivas y L. M. Patnaik. Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Trans Systems, Man, and Cybernetics*, 24(4):656–667, 1994.

R. Subbu y P. Bonissone. A retrospective view of fuzzy control of evolutionary algorithm resources. In *Proceeding of the IEEE Conference on Fuzzy systems*, pp. 143–148, 2003.

J. U. Sun. A taguchi approach to parameter setting in a genetic algorithm for general job shop scheduling problem. *Industrial Engineering and Management Systems*, 6(2):119–124, 2007.

G. Syswerda. Uniform crossover in genetic algorithms. In *Proceeding of the 3rd Int. Conf. on Genetic Algorithm*, pp. 2–9. Morgan Kaufmann Publishers, 1989.

H. Takagi y M. Sugeno. Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1): 116–132, 1985.

M. Tomassini. A survey of genetic algorithms. *Annual Reviews of Computational Physics*, 3:87–118, 1995.

C. F. Tsai y K. M. Chao. The contingent design for the optimal parameter settings of genetic algorithms. In *Proceedings of 12th International Conference on Computer Supported Cooperative Work in Design*, pp. 217–222, 2008.

A. L. Tuson y P. Ross. Adapting operator settings in genetic algorithms. *Evolutionary Computation*, 6(2):161–184, 1998.

P. Vajda, A. E. Eiben, y W. Hordijk. *Parallel Problem Solving from Nature*, chapter Parameter Control Methods for Selection Operators in Genetic Algorithms. Lecture Notes in Computer Science. 2008.

G. Venturini. Sia: a supervised inductive algorithm with genetic search for learning attribute based concepts. In *Proceedings European Conference on Machine Learning*, pp. 280–296, Viena, 1993.

L. D. Whitley, K. Mathias, y P. Fitzhorn. Delta coding: An iterative strategy for genetic algorithms. In *Proceedings of the 4th International Conference on Genetic Algorithms*, pp. 77–84. Morgan Kaufmann, 1991.

S. W. Wilson. Classifier system learning of a boolean function. Technical Report RIS-27r, Rowland Institute for Science, 1986.

D. H. Wolpert y W. G. MacReady. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

A. Wright. Genetic algorithms for real parameter optimizacion. In *Foundations of Genetic Algorithms*, volume 1, pp. 205–218. Morgan Kaufmann, 1991.

S. J. Wu y P. T. Chow. Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization. *Engineering Optimization*, 24(2):137–159, 1995.

L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.